REVIEWED! LATEST LINUX MINT 19

www.linuxuser.co.uk

Linux User & Developer™

THE ESSENTIAL MAGAZINE FOR THE GNU GENERATION

SYSADMIN SURVIVAL GUIDE

Smart scripting, clever auto-installs, better backups and more!

DISCOVER

DNA COMPUTING

Data that lasts for hundreds of years!

LEARN HOW TO

Make the most of KDE's

Plasma desktop

BOT WARS

AI is changing the face of security

48 PAGES OF **TUTORIALS**

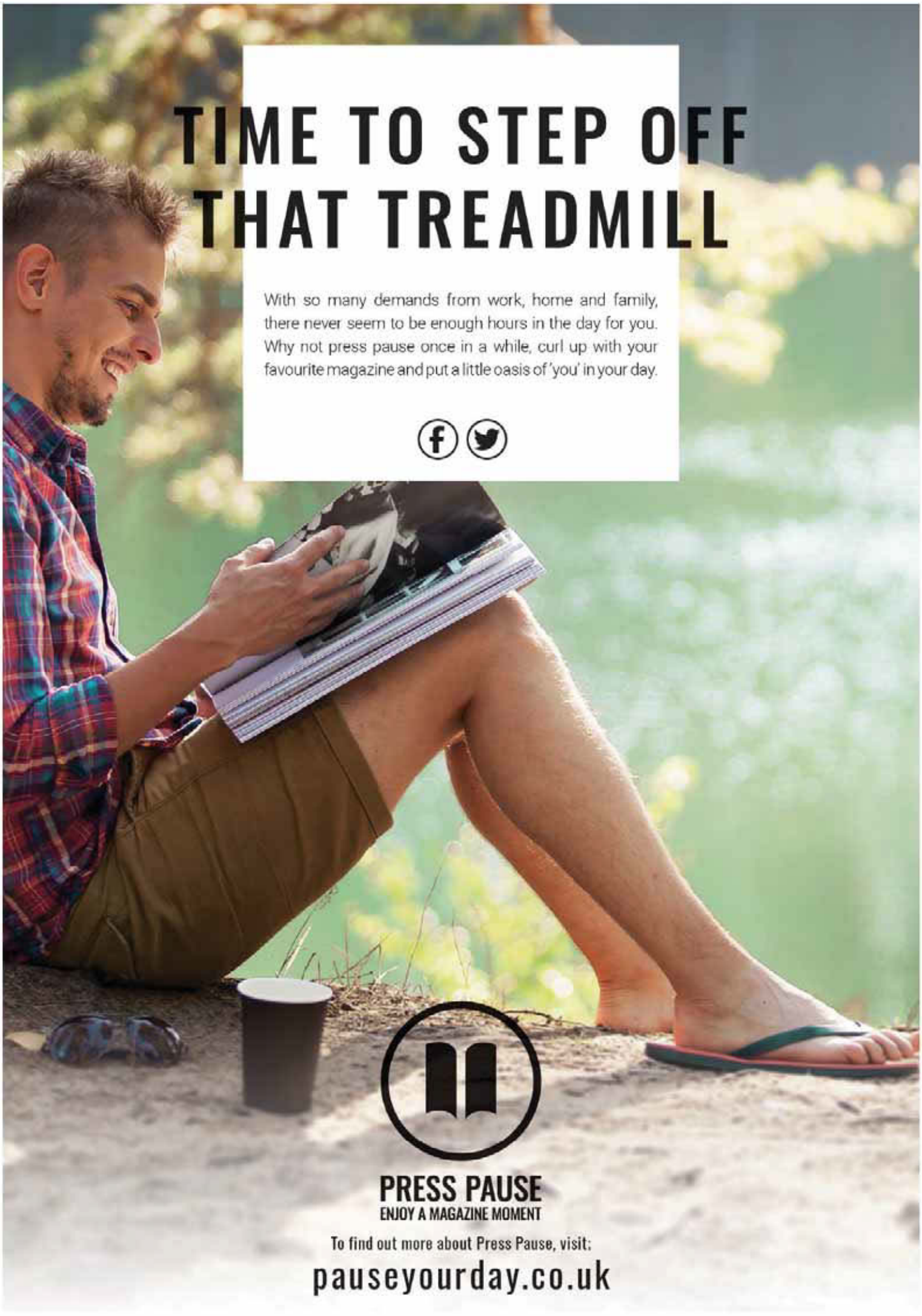
> Host a Git server + GitHub alternatives
> Make great GUIs > Code a YouTube clone

Recovery distros
We test the top distros for rescue and repair work

ePaper Pi projects
Build a digital name badge and more with the Inky pHAT

ALSO INSIDE
» Kernel news
» Pi fun: Build a game controller

FUTURE


TIME TO STEP OFF THAT TREADMILL
With so many demands from work, home and family, there never seem to be enough hours in the day for you. Why not press pause once in a while, curl up with your favourite magazine and put a little oasis of 'you' in your day.
PRESS PAUSE
ENJOY A MAGAZINE MOMENT
To find out more about Press Pause, visit:
pauseyourday.co.uk

Linux User & Developer
THE MAGAZINE FOR THE GNU GENERATION

Future PLC Quay House, The Ambury, Bath BA1 1UA

Editorial
Editor **Chris Thornett**
chris.thornett@futurenet.com
01202 442244
Designer **Rosie Webber**
Production Editor **Ed Ricketts**
Editor in Chief, Tech **Graham Barlow**
Senior Art Editor **Jo Gulliver**

Contributors
Dan Aldred, Mike Bedford, Joey Bernard, Christian Cawley, John Gowers, Toni Castillo Girona, Jon Masters, Cliff Newman, Paul O'Brien,Mark Pickavance, Calvin Robinson, Mayank Sharma, Shashank Sharma, Bill Thomas, Kevin Wittmer

All copyrights and trademarks are recognised and respected. Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Advertising
Media packs are available on request
Commercial Director **Clare Dove**
clare.dove@futurenet.com
Advertising Director **Richard Hemmings**
richard.hemmings@futurenet.com
☎01225 687615
Account Director **Andrew Tilbury**
andrew.tilbury@futurenet.com
☎01225 687144
Account Director **Crispin Moller**
crispin.moller@futurenet.com
☎01225 687335

International
Linux User & Developer is available for licensing. Contact the International department to discuss partnership opportunities
International Licensing Director **Matt Ellis**
matt.ellis@futurenet.com

Subscriptions
Email enquiries **contact@myfavouritemagazines.co.uk**
UK orderline & enquiries **0344 848 2852**
Overseas order line and enquiries **+44 (0)344 848 2852**
Online orders & enquiries **www.myfavouritemagazines.co.uk**
Head of subscriptions **Sharon Todd**

Circulation
Head of Newstrade **Tim Mathers**

Production
Head of Production US & UK **Mark Constance**
Production Project Manager **Clare Scott**
Advertising Production Manager **Joanne Crosby**
Digital Editions Controller **Jason Hudson**
Production Manager **Nola Cokely**

Management
Managing Director **Aaron Asadi**
Editorial Director **Paul Newman**
Head of Art & Design **Rodney Dive**
Commercial Finance Director **Dan Jotcham**

Printed by
Wyndeham Peterborough, Storey's Bar Road, Peterborough, Cambridgeshire, PE1 5YS

Distributed by
Marketforce, 5 Churchill Place, Canary Wharf, London, E14 5HU
www.marketforce.co.uk Tel: 0203 787 9001
ISSN 2041-3270

We are committed to only using magazine paper which is derived from responsibly managed, certified forestry and chlorine-free manufacture. The paper in this magazine was sourced and produced from sustainable managed forests, conforming to strict environmental and socioeconomic standards. The manufacturing paper mill holds full FSC (Forest Stewardship Council) certification and accreditation

All contents © 2018 Future Publishing Limited or published under licence. All rights reserved. No part of this magazine may be used, stored, transmitted or reproduced in any way without the prior written permission of the publisher. Future Publishing Limited (company number 2008885) is registered in England and Wales. Registered office: Quay House, The Ambury, Bath BA1 1UA. All information contained in this publication is for information only and is, as far as we are aware, correct at the time of going to press. Future cannot accept any responsibility for errors or inaccuracies in such information. You are advised to contact manufacturers and retailers directly with regard to the price of products/services referred to in this publication. Apps and websites mentioned in this publication are not under our control. We are not responsible for their contents or any other changes or updates to them. This magazine is fully independent and not affiliated in any way with the companies mentioned herein.

If you submit material to us, you warrant that you own the material and/or have the necessary rights/permissions to supply the material and you automatically grant Future and its licensees a licence to publish your submission in whole or in part in any/all issues and/or editions of publications, in any format published worldwide and on associated websites, social media channels and associated products. Any material you submit is sent at your own risk and, although every care is taken, neither Future nor its employees, agents, subcontractors or licensees shall be liable for loss or damage. We assume all unsolicited material is for publication unless otherwise stated, and reserve the right to edit, amend, adapt all submissions.

FUTURE
Connectors. Creators. Experience Makers.

Future plc is a public company quoted on the London Stock Exchange (symbol: FUTR)
www.futureplc.com

Chief executive **Zillah Byng-Thorne**
Non-executive chairman **Peter Allen**
Chief financial officer **Penny Ladkin-Brand**
Tel +44 (0)1225 442 244

Welcome

to issue 194 of Linux User & Developer

In this issue

» Sysadmin Survival Guide, p18
» Power User's Guide to Plasma, p58
» Computing with a Double Helix, p64

Welcome to the UK and North America's No.1 Linux and open source magazine.

I was chatting to Michael Meeks at the Collabora offices recently (see p12) and something struck me about his response to the GitHub acquisition. He was less interested in Microsoft as the buyer and more focused on the new CEO, Nat Friedman. Friedman has made serious contributions to the open source movement and he's someone – along with Miguel de Icaza, a founder of GNOME – who's attributed with encouraging Microsoft to open-source projects such as .NET. So maybe we need to give GitHub a chance? However, we know some of you are considering your Git hosting options, so flip to p34. Just to emphasise we're not being mean, we've got a tutorial on Azure Powershell as well on p40. This issue, we also start a new Go programming series (p52)!

You've requested more sysadmin and explainer articles, so we've covered the Dark Art in our main feature (p18) and written a KDE Plasma guide (p58). For our third feature, we've a surprisingly near-future look at DNA computing (p64). Enjoy!

Chris Thornett, Editor

Get in touch with the team:

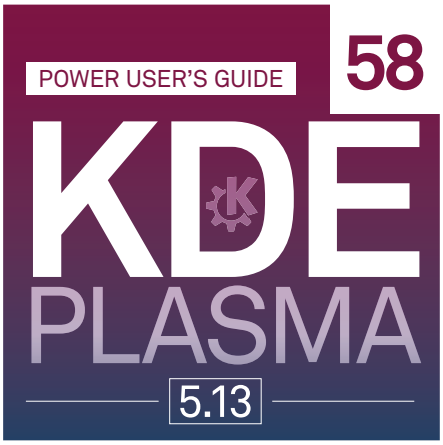
linuxuser@futurenet.com

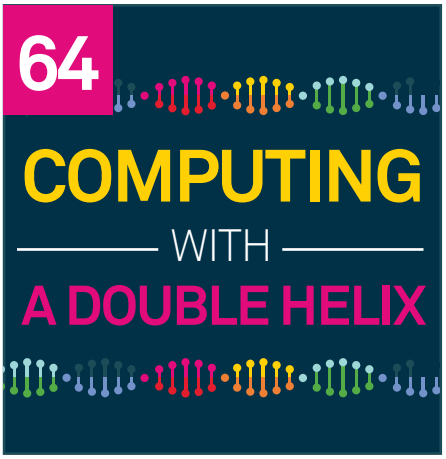
Twitter: @linuxusermag

Facebook: facebook.com/LinuxUserUK

Filesilo: filesilohelp@futurenet.com

Contents





OpenSource

06 News
SUSE has been bought out, and there's yet another CPU security flaw

10 Letters
Don't hold back – we can take it!

12 Interview
Michael Meeks of Collabra Online talks about his open source journey

16 Kernel Column
Jon Masters on the latest updates

SpecialReport

32 Machine Learning in Security
We visit the InfoSecurity Europe show to find out how ML is driving security

Features

18 Sysadmin Survival Guide
New to system administration, or need to know what best practices might get you the job? Check out our complete guide to every aspect of the role, from mastering network essentials to cloud and virtualisation techniques

58 Power User's Guide to Plasma
The latest KDE Plasma desktop environment is faster, more beautiful and more mature than ever before. Our power user's guide will get you up to speed with the latest release

64 DNA Computing
Organic DNA may be the solution not only to the need for ever-increasing computer power, but could also provide vast amounts of storage space

Tutorials

34 Essential Linux: Git
Part four of our series on Git explains how to host your own server

40 Manage Azure resources
Windows' PowerShell in Linux is particularly useful for handling Azure services. Find out why and how

44 Computer security
Learn how to use public REST APIs during your security engagements

48 Python: Kivy
How Kivy can give your code a more friendly user interface

52 Programming: Go
Use Google's Go language to create a simple YouTube-style video site

LinuxUser & Developer
Issue 194
July 2018
facebook.com/LinuxUserUK
Twitter: @linuxusermag

FileSilo
94 Free downloads
We've uploaded a host of new free and open source software this month

Practical Pi

72 Pi Laser Tag
Using a couple of Pi Zero W machines, some 3D printing for the 'guns' and a simple desktop server, **Terran Gerratt** created his own laser tag game

74 Create a Gaming Controller Prototype for the Pi
It's remarkably simple to create a basic console-style controller for the Raspberry Pi, with some breadboard and a few push buttons connected to the GPIO ports. We show you how

76 Getting Started with the Pimoroni Inky pHAT
Pimoroni's beautiful 212x104-pixel, three-colour ePaper device is ideal for physical displays of data, as we explain in this guide

Reviews

81 Group test: Recovery Distros
ALT Linux Rescue, MorpheusArch Linux, SystemRescueCd and Ultimate Boot CD all promise to rescue and repair a Linux installation with the minimum of hassle

86 Acer Chromebook Spin 11
Does the Acer Chromebook Spin 11 have enough oomph to function as a modern notebook?

88 Linux Mint 19 Cinnamon 'Tara'
This latest LTS release of the hugely popular 'Linux beginner's distro' has a lot to live up to

90 Fresh FOSS
Cinnamon 3.8.4, Crystal 0.25, uftpd 2.5 and Stacer 1.0.9 reviewed

Back page

96 Obnoxious
Another intriguing short story from sci-fi author Stephen Oram

SUBSCRIBE TODAY
Get a free high-speed USB charger when you subscribe!
Turn to page 30 for more information

BUSINESS

Private equity buys SUSE for $2.5 billion

SUSE was the only profitable business for former owner

British technology company Micro Focus International has agreed the sale of its business-focused SUSE Linux operating system to private equity firm EQT Partners, for $2.53 billion (around £1.9 billion).

Now in its 26th year, SUSE has been under the wing of Micro Focus International since 2014, when the company merged with Attachmate, which had previously purchased SUSE as part of Novell in 2011. SUSE had been part of Novell's plan to get its business back on track with an acquisition in 2003; Novell's technologies will remain with Micro Focus.

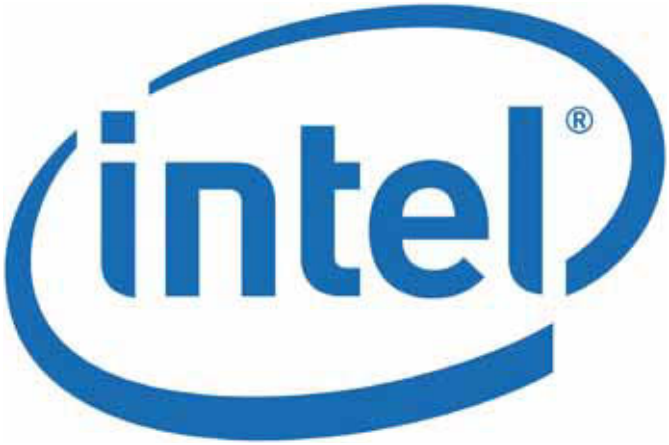
Originally developed in Germany, SUSE has long seen itself as a competitor to Red Hat Linux. Just as Red Hat Linux has the free Fedora, so SUSE has openSUSE. As with Red Hat Linux, SUSE generates revenue from technical support (PC and server), hosting, virtualisation, systems applications products, cloud apps and storage, certification and more. Under Micro Focus International, SUSE has had a lot of success, charting sales of $320 million in the 12 months to 31 October 2017, with 1,400 employees around the globe. This is due to its position as an enterprise-grade provider of open source defined infrastructure offering application delivery solutions for both local and cloud-based work.

Johannes Reichel is a Partner at EQT Partners says that the company was excited to partner with SUSE's management as an "attractive growth investment opportunity", he said. "We were impressed by the business' strong performance over the last years as well as by its strong culture and heritage as a pioneer in the open source space."

Following the transaction, the current CEO of SUSE Nils Brauckmann will retain his position. Describing news of the sale as "an exciting day in SUSE's history", Brauckmann emphasised that it means that SUSE can "become a full independent business". He added: "The next chapter in SUSE's development will continue, and even accelerate, the momentum generated over the last years. Together with EQT, we will benefit both from further investment opportunities and having the continuity of a leadership team focused on securing long-term profitable growth combined with a sharp focus on customer and partner success."

Above SUSE has gone from shipping its first distro on 40 floppy disks to being acquired by EQT for $2.53bn

The sale is subject to approval from Micro Focus International's shareholders, and the usual regulatory bodies. How long SUSE remains under private equity ownership beyond this is anyone's guess.



SECURITY

New CPU flaw hits older Linux systems

Xen Project among casualties, with a fix issued in new patch

Hot on the heels of the Meltdown and Spectre CPU bugs comes Lazy FPU, a newly discovered bug that enables floating point data to be misused. But while this is a vulnerability that can be exploited in a similar way to its infamous predecessors, it is one that can be patched far more easily, and without the performance hits that patching Meltdown and Spectre incurred. Indeed, mitigation of this bug actually improves system performance.

Better still, Lazy FPU doesn't affect modern versions of Linux (kernel 4.9 onwards), Windows or the most recent OpenBSD and DragonflyBSD. However, the Linux team is reported to be working on mitigation measures for pre-4.9 kernels.

The bug – discovered by chance during ongoing investigation into Spectre-Meltdown – is found only in Intel Core (Nehalem and later) and Intel Xeon processors, and can potentially allow malicious actors or malware to grab sensitive data from the maths processing units. This data is held in FPU (floating point unit) registers, and could be subverted to uncover cryptographic keys, thereby potentially decrypting data used in secure applications and websites.

Intel's summary of the bug says: "System software may utilise the Lazy FP state restore technique to delay the restoring of state until an instruction operating on that state is actually executed by the new process. Systems using Intel Core-based microprocessors may potentially allow a local process to infer data utilising Lazy FP state restore from another process through a speculative execution side channel."

It isn't just standard desktops and servers that are at risk from this flaw. Virtualisation project Xen's hypervisor has also been hit, with the following explanation: "An attacker can read x87/MMX/SSE/AVX/AVX-512 register state belonging to another vCPU previously scheduled on the same processor. This can be state belonging a different guest, or state belonging to a different thread inside the same guest."

Suggestions that VMWare has also been hit by the bug have been denied, however, with the virtualisation giant admitting that only a few of its Linux-based virtual appliances are affected.

At this stage, the flaw is considered to be safe from exploitation, but patching is always safer than taking a chance.

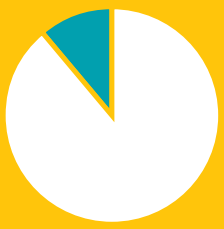
DISTRO FEED

Top 10

(Average hits per day, 30 days to 29 June 2018)

1.	Manjaro	▼	4101
2.	Mint	▲	2508
3.	elementary	▼	1483
4.	MX Linux	▼	1349
5.	Ubuntu	▼	1191
6.	Debian	▼	1029
7.	openSUSE	▼	775
8.	Fedora	▼	696
9.	Solus	▼	691
10.	Lite	▼	609

This month



In development (2)
Stable releases (16) Falling distro hits might suggest Linux users are satisfied with their distro... but that doesn't really explain why Mint bucks the trend.

Highlights

Peppermint
Based on Lubuntu, Peppermint is fast and lightweight, and uses a hybrid LXDE/Xfce desktop environment, mixing lxsession with Xfce's panel.

Deepin
Formerly Linux Deepin, this distro is now Debian-based (replacing Ubuntu). The distro aims to be a good Windows alternative.

Devuan GNU+Linux
Forked from Debian, this project aims to be an alternative, without the issues surrounding systemd. Devuan GNU+Linux features the Xfce desktop.

Latest distros available:
filesilo.co.uk

RASPBERRY PI

Raspbian update brings new Pi features

Features first-boot setup wizard, new PDF reader and app store

Fresh from the Raspberry Pi Foundation is a new version of the Debian-based Raspbian desktop, optimised for the Raspberry Pi 3 B+ and boasting a number of new features. Primary among these is the 'first-boot setup' wizard, a long-overdue inclusion that guides the user through setup options, as with most other Linux distributions.

Adding the ability to set location, check for updates, add a password (ending, at last, the security issue of the default username and password), and choose Wi-Fi networks, this feature is automatically disabled for subsequent boots. It can, however, be accessed at any time using the `piwiz` command in the terminal.

Following the launch of the Raspberry Pi in 2012, the short-lived Pi Store provided an easy way to install approved software. Since then, more and more software has been bundled into the Raspbian OS. "We've always included these applications in our standard image, as people might never find out about them otherwise," says Senior Principal Software Engineer Simon Long. However, he added that "the applications perhaps aren't all of interest to every user." As a result, the Recommended Software tool has been included, partially to reduce the file size of the Raspbian disk image, but also to minimise menu clutter.

> "The Recommended Software tool helps to reduce menu clutter"

Welcome to Raspberry Pi
Welcome to the Raspberry Pi Desktop!
Before you start using it, there are a few things to set up.
Press 'Next' to get started.

Above It's now even easier to set up a Linux distro on your Raspberry Pi

Along with these notable improvements, the new version of Raspbian (still based on Stretch) also includes a new PDF reader called qpdfView, as well as improved keyboard volume support, and various other bug fixes and UI tweaks. Chromium 65 is included, along with some new browser-based help pages.

The x86 version of Raspbian has also been updated; both can be downloaded from **www.raspberrypi.org/downloads**, or you can update your current system from the terminal if you prefer.

DISTRO

Canonical shares desktop metrics

Canonical has revealed that 67 per cent of users opted-in to the collection of desktop metrics when installing Ubuntu 18.04 LTS.

Controversial upon its announcement, the Ubuntu Report tool was integrated into setup to record hardware and localisation options. Intended to help Canonical understand what systems are running Ubuntu – and how it can improve the OS – the decision to include the tool has reportedly turned long-term Ubuntu users away from the distro. That's despite assertions from Canonical that the reporting is not an actual privacy breach (no user-identifiable information is shared), and the fact that all data is available to review by the user before it's even sent.

While some of the information revealed is a little dry, Canonical has been able to get an idea of the current state of Linux PCs and laptops. Unsurprisingly, Ubuntu is used all over the world (this information was gathered via timezone selection – no IP addresses are stored), with the BRICS nations of Brazil, Russia, India, China and South Africa particularly keen on the operating system. Meanwhile, 90 per cent of users choose to download updates during installation, with 53 per cent requesting restricted add-ons. Perhaps worryingly from a security standpoint of view, 28 per cent choose automatic login.

Metrics also reveal that systems with 4GB and 8GB RAM are most common, as are single CPUs. Single GPUs also outweigh dual-CPU systems, but things are a little more complicated when it comes to hard disk storage. Canonical found that 21.89 per cent of users installed Ubuntu 18.04 LTS to a manually created partition table. This single revelation has already prompted Canonical to investigate how to simplify the process for Ubuntu 18.10.

OPINION

Demystifying Kubernetes

Making it work for you can take some effort, but it's worth it

There's unquestionably a heap of confusion around getting started with the now-dominant container orchestrator, Kubernetes. It's no longer a particularly new piece of software, but I think it's safe to say that when it was first released it appeared relatively arcane.

The confusion is certainly lessening and exists for a number of reasons. Don't get me wrong; the Kubernetes community has done a great job in documenting the application, but the barriers to entry definitely still exist for some new users.

I recently saw a how-to article offering people with minimum Docker experience a way into Kubernetes – and that leads me to issue number one. The first prerequisite is having had a 'Eureka!' moment of how much containers change the way software releases work (and how they reduce night terrors when running infrastructure).

This comes about after building and breaking Docker containers for a while and then muttering "Wow". I remember having this moment myself, even though I had used containers on a production estate in 2009. Granted, those containers weren't used for CI/CD pipelines, but you do need container experience from the off.

Secondly, there's no getting away from the fact that Kubernetes can grow monstrously complicated. I am referring to adding persistent storage, multiple cloud regions, several applications and extra redundancy to an otherwise default install. Integrating all the required bells and whistles to provide production-viability can be mind-boggling. All this takes practice and, frankly, usually lots of trial and error.

Factor in the initial stumbling blocks with a number of moving targets, and setting up Kubernetes is no small endeavour. The targets that I refer to are there thanks to the (fantastic) rate of innovation that Kubernetes undergoes. On a regular basis, features become deprecated, syntax is no longer valid and brand new (necessary or nice-to-have) add-ons require themselves to be learned promptly.

Compound those challenges with sometimes even more confusing developments in the software space surrounding the third-party tools that you integrate with Kubernetes, and I'm sure some people will agree it's a bit like running at full speed on a treadmill.

One saving grace – and again, don't get me wrong, as I am still a huge advocate of Kubernetes and believe it is the future for many reasons – is a lovely installation tool called Minikube. It takes almost all of the headaches out of the tricksy and frankly bewildering installation process when you first try a multi-region cloud installation. Of course the trade-off is that you only have a local copy of a Kubernetes cluster. For most people, however, I suspect that will whet their appetite enough.

The mighty Minikube punches well above its weight and also conveniently installs itself into a local virtual machine to save headaches with the mountain of config and post-install cruft required.

Indeed, if you've got a laptop with the right CPU extensions for virtualisation – which are actually optional, but useful to have – and you haven't tried Minikube yet, you might just find your calling as a Kubernetes developer by visiting the Kubernetes site at **http://bit.ly/lud_minikube**. I can't recommend it enough. The path to enlightenment with Minikube is without question a shorter one than with other options. Try launching some simple services as containers once you're up and running and I'm sure you will rapidly become a convert to this way of working.

> "The first prerequisite is having had a 'Eureka!' moment of how much containers change the way software releases work"

Once you are comfortable with running a daemon set or two, a config map here and an ingress controller there, you could happily progress to other installation methods. My personal preference for your next step would be to try an AWS (Amazon Web Services) installation using the details highlighted on 'Picking The Right Solution' at the Kubernetes' website, **http://bit.ly/lud_solution**.

I would opt for a cloud build so that mistakes can be rectified promptly and new builds tried again for trial and error. That ostensibly sounds simple, but given the list (see right) showing actively maintained Kubernetes solutions, it will be worth reading up on them first to assess their suitability for your needs.

Once you're up and running you'll potentially benefit from resilient and scalable infrastructure running on different continents, which is hard to rival. And, ultimately, you'll appreciate what all the fuss is about.

Chris Binnie

Chris' book *Linux Server Security: Hack and Defend* teaches you to make servers invisible and mitigate attacks. See **www.devsecops.cc**

QUICK LIST

Kubernetes solutions

- AWS
- Azure
- CenturyLink Cloud
- Conjure-up Kubernetes with Ubuntu on AWS, Azure, Google Cloud, Oracle Cloud
- Google Compute Engine (GCE)
- Kubermatic
- IBM Cloud
- Madcore.Ai
- Rancher 2.0
- Stackpoint.io
- Tectonic by CoreOS

COMMENT

Your letters

Questions and opinions about the mag, Linux and open source

TOP TWEET

@Pepulani7
@LinuxUserMag really glad Toni Castillo Girona dedicated an entire article on Metasploit shellcode in @LinuxUserMag issue #192. Been really looking forward to this!

Right Microsoft is trying to bolster its developer products and support, and GitHub was a good fit for that. The question will be how it approaches making money from its $7.5 billion investment

GitHub hokum

Dear LU&D, I read with interest your little editor's introduction column in last issue (**LU&D193**) about some people in the Linux community moaning about the GitHub acquisition. I recall you suggesting, along with some neckbeard-types, that handing GitHub to an independent foundation would save the company from being tainted by Microsoft and the company's evil plans to destroy Linux from the inside. Everyone needs to move on!
Name supplied

Chris: Hey, neckbeards keep you warm – and anyway, that isn't quite what I wrote. The point I was making was that because GitHub has become such a central part of the open source community and is the way that tens of thousands of projects share their code and collaborate, it has been felt by some that it needs to be impartial. A foundation is a clean way to achieve this, but it's unlikely to happen when doing so involves ignoring Microsoft's $7.5 billion investment.

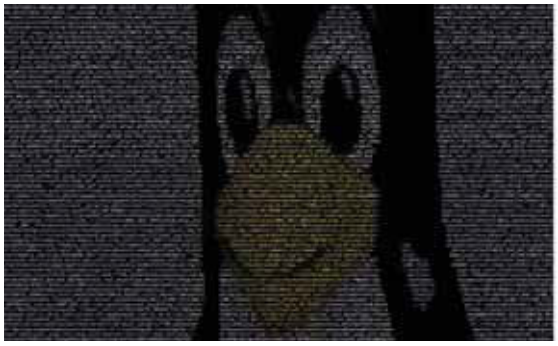
What the GitHub acquisition *has* done is highlight the clear divide in the Linux community, which boils down to the differences between the ideology of the free software movement and the emergence of the pragmatic open source movement. I can see both sides of the argument: Those with long memories will struggle to forgive Microsoft's concerted attempts to kill Linux. But – as Collobora's Michael Meeks commented to me for this issue's interview – companies aren't a single human being (even if seen as a legal entity) and there have been Microsoft employees patiently encouraging the Redmond giant to embrace open source development for years.

Yes, Microsoft's change is motivated by pragmatism; it knows that to compete it needs to embrace open source, but the point is that it has changed and isn't vaguely interested in destroying Linux, especially now it's using slices of Linux and numerous open source projects for its own products!

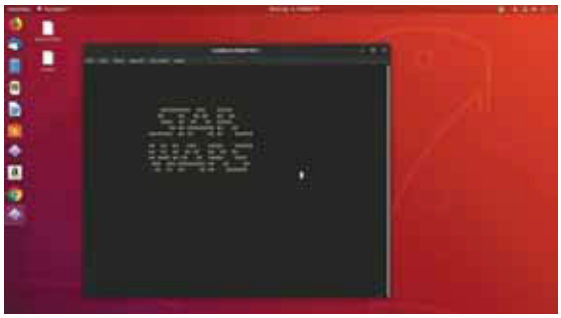
Of course, there's a danger that a new Microsoft CEO might have different ideas or that, in the future, open source software falls out of favour with the business world – but that seems unlikely, as once it has been embraced the open source model just can't be beaten. Ultimately people will decide, and many are already moving to GitLabs *et al*, but it seems deeply mean-spirited to beat Microsoft over the head for something it *might* do in the future.

CLI Anonymous

Dear LU&D, Hello, my name is Chuck and I'm scared of the command line. I've been terrified of the CLI for 10 years now, but I have become increasingly worried about this ailment since I picked up your magazine from my local supermarket two years ago. I see myself as a novice Linux user because of this fear, and I blame it on Windows, as I didn't know or need to use it for that operating system. I must confess that most of what you



www.100MB.nl



Above The command line isn't a scary place. You can even watch a text-based version of Star Wars A New Hope when you're bored: type `telnet towel.blinkenlights.nl`

publish zips straight over my head, but I have learnt a lot and make a point to pick up a copy when I see it stocked. Keep up the good work!

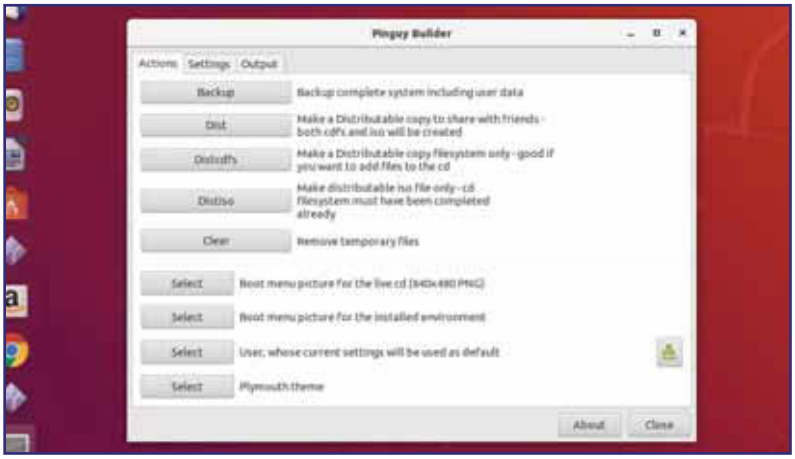
Chuck Hellmann

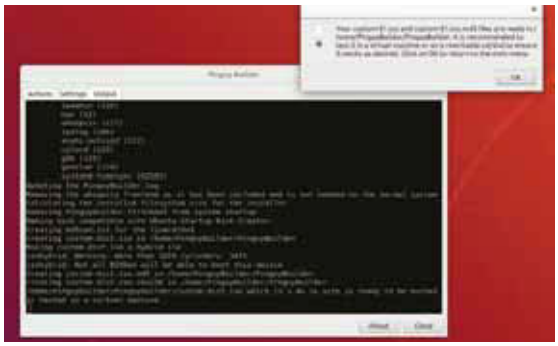
Chris: Thanks, Chuck. You must not fear. Fear is the mind-killer. Fear is the little-death that brings total obliteration... Yes, that inky emptiness that the window presents to you can be daunting. I think that's probably the hardest part for many users as it's very bare and the onus is on the user knowing what to type. But it's a question of being a little curious and using commands regularly enough to build familiarity with the syntax.

For instance, we spend so much time each month testing ISOs – the images we create to put on the cover disc – we've caught ourselves typing away in our sleep trying to duplicate imaginary files! Of course, we use commands a lot in the magazine and if you want to learn more about any of them you can try a quick Google search, or you can open up a terminal and type `man [commandname]`. The `man` command is short for 'manual' and you'll see the documentation for each command. We've been talking about devoting a tutorial to highlighting a specific command or utility each issue, and it'd be interesting to hear from readers if they thought that would be a good idea.

Duplicating distros

Dear LU&D. I would love it if you would make a future issue that covered how to make your own Linux distro and/or desktop environment. Not necessarily writing one from scratch with C or something, but creating bootable media of your current system and making it installable. I would create a custom clone of


Pinguy Builder
Actions
Settings
Output
Backup
Backup complete system including user data
Dist
Make a Distributable copy to share with friends - both cd's and iso will be created
Distsfs
Make a Distributable copy filesystem only - good if you want to add files to the cd
Distiso
Make distributable iso file only - cd filesystem must have been completed already
Clear
Remove temporary files
Select
Boot menu picture for the live cd (640x480 PNG)
Select
Boot menu picture for the installed environment
Select
User, whose current settings will be used as default
Select
Plymouth theme
About
Close



Left There are lots of ways to copy your current distro to create a live version, and Pinguy Builder is one that makes the process very simple

Ubuntu, but with WPS office instead of LibreOffice, and a custom desktop that I built with Openbox. If there is already an issue covering this, I would appreciate the issue number. Thank you!

Matt W

Chris: Good suggestion, Matt. Unfortunately, we have a limited archive for seeing if we have done such a thing before. I imagine that we must have done, though. A number of the big distros have builders for making a custom version, but in regard to copying your existing system, there's PinguyOS's Builder (**http://bit.ly/PinguyOSBuilder**). This is actually a fork of remastersys, which was discontinued in 2013. It's a GUI-based system and very straightforward to use, although we can't guarantee that it will work; we've tried it on Ubuntu 18.04 and it did the trick. Alternatively, you can do all this malarkey manually by using a downloaded ISO of the distro you want to use, squashfs and imkisofs. Perhaps we need to do a tutorial on it?

INTERVIEW COLLABORA

Meeting Mr Productivity

We travelled to the Collabora offices in Cambridge to visit Michael Meeks, a veteran open source advocate and developer

Michael Meeks
Michael is general manager at Collabora Productivity, part of Collabora, an open source consulting business that supports many projects including LibreOffice, Linux kernel, Wayland and more.

Meeks is a mild-mannered heavyweight of the open source world, who has worked tirelessly for the GNOME project for many years and was an early employee of Nat Friedman and Miguel de Icaza (one of the founders of the GNOME project) when they set up Ximian to develop the Ximian Desktop. We interviewed Meeks in Collabora's Cambridge offices. For Meeks, Collabora is the culmination of a number of moves in the open source software business which saw him join Novell when Ximian was acquired in 2003, and work at SUSE until Collabora was formed.

Tell us about your background and how you got into open source...
Sure. My mother taught me to program when I was very small, which was fun. It was a BBC Micro and I got really into it. I loved to write games. In my gap year I became a Christian, and it turned out that all the games I was writing used stolen compilers and stolen operating systems. It was all a bit risqué. So in the end, I switched to using this Linux thing, which at the time was absolutely terrible. It literally killed my hard disk the first time I installed [it]. There were no graphics worth using. There was no way you could be writing games.

Above Michael Meeks has come a long way from writing games on the BBC Micro

Tell us more about what you were doing on the GNOME side of things.
My first contribution was for GCC [the GNU Compiler Collection]. I was just irritated that there was a sort of undefined behaviour thing happening but the compiler wouldn't tell you this. You turn all warnings on and nothing would go, 'Hey wait a minute, this could do almost anything.'

So I implemented a patch for this; sent it to the GCC people; and I didn't get a reply for like two years. Two years later, I got a thing saying, 'Oh, I was looking into fixing this. I discovered there's a patch on the mailing list that does exactly this. There's really no problems with it. So I just fixed up the help and merged it', so that was not really a very good validation of the model. So I moved on. I installed GNOME, and I started playing with a Mahjong game, making it solvable and so on. And then I got into Gnumeric with Miguel [de Icaza] and hacked a lot on that, doing file-format reverse engineering. The first Excel filter is really open source – we could grab binary files and import them. And then the export filters were even more exciting, because it was before security became an issue for Microsoft, so there was no real auditing or fuzzing on their filters.

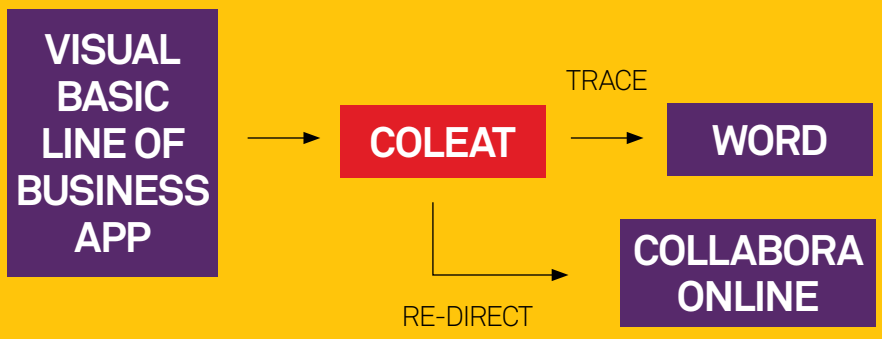
The typical export test was: look at the diff to their file [which prints the lines that are different] and then try to reduce it, and see if it crashes when it loads in Excel and see if it crashes after playing with it for a while. Because often you'd corrupt the heap, or something really bad would happen and subsequently you would crash – caused by your bad file format. So yeah, there were all sorts of horrors under the hood. The world has got much more secure in recent times. Now it tends to crash on import or tolerate bad input much better than it did in Microsoft Word and, of course, LibreOffice too.

So going back to being founded out from SUSE, which part or projects do you cover? Collabora Office and Collabora Online?
Collabora Productivity does LibreOffice-based stuff, essentially. So we have Collabora Online [an online productivity suite], we have Collabora Office [a Microsoft Office alternative]. We have a few other things around that can lead to the various bits of those products. But essentially everything

QUICK GUIDE

Easy VBA migrator

A key feature of Collabora Office 6.0, which was released in June, is a new interoperability tool called COLEAT (Collabora OLE Automation Tool) which can be used as a drop-in replacement for legacy systems that use components such as VBA scripts. This is enabling many organisations to migrate to open source. As an example, Meeks says Ulster Hospital in Belfast, Northern Ireland, had a problem with a system called Patient Centre, created by Computer Services Corporation (CSC). The system was developed in Visual Basic 6, explained Meeks: "You can no longer really buy the compiler for VB6. You need a professional edition, and it's hundreds of dollars on eBay as Microsoft doesn't sell it any more. But this thing essentially screen-scrapes an HPUX mainframe service that sits at the back-end and it then provides all sorts of things for managing patient data. As part of that, of course, it integrates with Microsoft Office." COLEAT has enabled the hospital to migrate away, which Meek says, amounts to migration of 8,500 people with "several thousand" now using Collabora Office.



Above The API development work for COLEAT has opened the door for many organisations to consider an open source alternative to systems that use legacy VBA scripts and unsupported versions of Visual Basic

we do is open source, so it's all contributed back to the community. And yes, we work very heavily on LibreOffice as well, obviously. We have two directors on the board there, and a membership committee member, and lots on the advisory board. We sponsor the conferences and the project. We've over a third of the commits to the code, and so on. Yeah, a huge investment there from Collabora.

With Collabora Office 6.0, you're calling it the migrator's choice.

Yes, migrator's choice for Collabora Office 6.0 – there's lots of good things in there. Lots of the work we do is consultancy work, but I think when we can invest in things ourselves, it's nice to do things that help people into the project.

One of the things there, for Windows users particularly, is that they suffer from these nasty little applications that people write using Visual Basic. Like parking ticket systems – every council somewhere is giving out parking tickets and every police department has one of these things. It's inevitably some bespoke tool that's written with a little database, a bit of Visual Basic.

To print your parking ticket, there's some kind of Microsoft Word integration and mail merge-y type thing at the back-end. Some of those things are so old, there's no source code for them any more. They're just gone. They cost money to create, so why would you pay more money? The problem is just the sheer number of diverse little things like this. It makes it hard for people to move to LibreOffice. So we have this interoperability API that allows us to look like Microsoft Office. These tools can run and not even be changed. You can drop Collabora Office in, and it'll just work nicely. That's called COLEAT – a Collabora OLE automation tool (see Easy VBA Migrator, above).

You can trace these things, and you can see what they're doing. You can actually see inside your application for the first time and see what silly stuff it's up to and replace it with Collabora Office.

Are there any features that you have on a list to target to make transitions even easier?

In terms of other features to target for people like that, I think interoperability is traditionally the bread-and-butter, the file format interoperability. And I think PowerPoint has been quite weak. We've done a whole chunk of work in 6.0 just going and finding all of the Microsoft templates; making sure that they import nicely; and that you don't get bits that we don't understand of the document. We export good open XML. Line spacing is fixed. There are a few self-inflicted wounds that we've fixed. I hope that it gets a lot better in 6.02.

But yes, just across the board, loads and loads of work in inoperability. SharePoint integration is something we've done quite a lot – IQY files, which people use for lists in their SharePoint service. We can download and render those on Linux as well, which is cool.

They have sort of queries built in. You get a spreadsheet view out of SharePoint giving you stuff. Yeah, just all sorts of things, richer AutoText. The problem partly with this is that it's hard to come

> "I switched to using this Linux thing, which at the time was absolutely terrible. It killed my hard disk"

Collabora Online

up with a coherent narrative for the whole thing, because the narrative is really 300 people fixing lots of things for a year all over the place, and making so many things better. Eight million lines of code. It's quite a lot there to improve and fix all over the place.

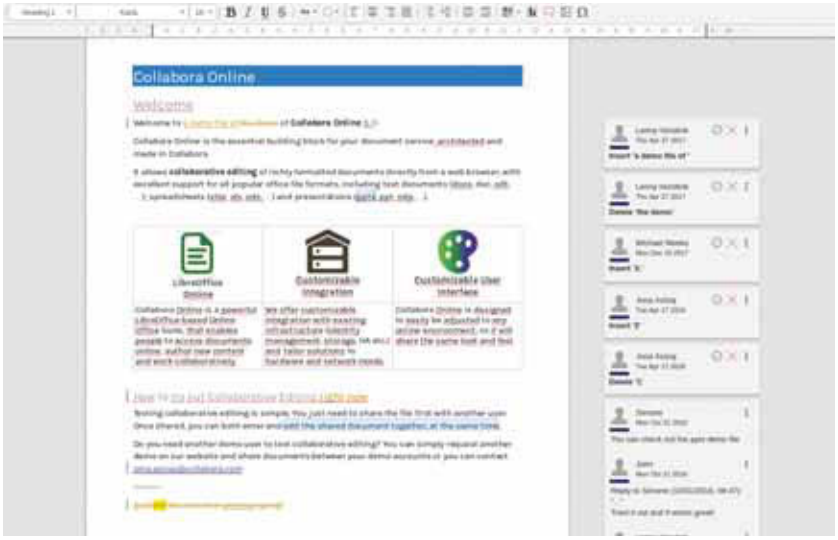
What are the security issues that you're experiencing with Collabora Online?
Security's an interesting challenge, obviously. Having said that, when you look at other people in the market, they're using LibreOffice in a much less secure way. I can give you several large companies and brand names that use LibreOffice in a relatively unusually insecure way. The first thing we do is just improving Collabora Office, which is running on people's PCs, not in a sandbox everywhere, and they're getting documents all the time – downloading them. So there's a big effort in the LibreOffice project, primarily driven by Red Hat, to fix this.

We have Coverity, which is a static-checking tool, and Coverity Scan (**https://scan.coverity.com**). Our score on that is approximately 0.00 bugs per 100,000 lines of code that they can find. The numbers there are very low and are kept very low. It's a beautiful thing, because as you type some bug into your code and commit it to Git, within a week you have an email saying, "Oh, this code is not very good, because it fails all these checks." So as we develop, we see all these problems, and they're fixed before they ever get out of the project.

So Coverity is very good. We have a lower score than Linux Kernel, in theory, which is quite funny.

Above Collabora Online is an editing suite for enterprises that has a per-user subscription, but you can try CODE (Collabora Online Development Edition) if you have your own Nextcloud or ownCloud. According to Meeks, this has already seen about 5.5 million Docker downloads

Below Collabora Online puts LibreOffice in the cloud for enterprises and integrates with Nextcloud



But of course, there are still plenty of issues in it. So we then have a thing called a crash-testing cluster. We have 105,000 bug documents, and we load all of those; we save them in each of the file formats we can export to; and we look for hangs, crashes, layout problems – all these sorts of things.

That, again, finds loads of bugs. It's really, really good. It takes about three days to run, so it's a big old test set. And of course, all of these are known bug documents, so they are ones that previously had problems, which is just a brilliant set. We scrape freedesktop, the GNOME bugs, the KDE bugs, the Launchpad, Novell... we cargo-cult bad documents from everywhere. It's really good. So if you know where bad documents lurk, let us know.

A graveyard of documents?
Yeah, really it is! In recent times, the Linux Foundations – after the SSL Heartbleed thing – set up OSS-Fuzz (**https://github.com/google/oss-fuzz**). OSS-Fuzz is awesome. Previously, we used to use AFL, which is the American Fuzzy Lop, a fuzzer (**http://lcamtuf.coredump.cx/afl**). These things use genetic algorithms, essentially, to 'breed' corpuses of documents with each other. They also watch the code it executes to try to get it down new branches in the code. These are frighteningly good tools. If people are paying large QA departments, they should be automating them out, because AI is coming to QA, and it is AFL and fuzzy.

Really, it's quite frightening. Basically, we take this 100,000 corpus of files; AFL will run through all of these things and go, "Yeah, these are good documents, but really, these are the documents that you want, that exercise the widest things. So let's just do it on these hundred documents instead." It will go away, and it will breed these documents with each other. It thrusts those all through LibreOffice and sees what happens. It follows, as I say, the branches, and it tries to get good coverage. It will then report the crashes to you, but better than reporting the crashes, it will also automatically minimise the crash.

So typically, what you find is there's some giant file which covers loads of code, and it finds a crash somewhere. But wouldn't it be nice if you could have a much smaller file that actually causes that crash? So it'll then minimise the file by chopping bits out of it and breeding again to try to get a very small thing.

And it's amazing. When you see, for example, the HTML filter – it's a human-readable file format, so you can see this huge file. It's managed to find some HTML that crashes the import filter in some way. And then it'll crunch it down to something like 64 bytes. This has found bugs that have been there for 20 to 25 years, never touched by anything else. So that's really good.

We were keeping ahead of the other baddies out there who would find things and report them. But when Google introduced OSS-Fuzz, we now have a thousand-machine cluster that runs this incredibly wide and that got us massively ahead. The slightly frightening thing is that you can run AFL and it makes progress through all these branches for weeks, and you think, "Ah, we've fixed them all." Then you throw it on the 1,000-CPU cluster and bang, you've got another one-a-day for the next week.

[There's also] a nice tool which basically simulates keyboard input – so you can turn your keyboard input into LibreOffice into a file format. This found, within a few hours, one of our most vile problems in Impress, where we had a copy/paste that would sometimes crash, but no one could reproduce it. This found it and let us fix it, just like *that*, with a reproducible test that you can push through. It's amazing tooling.

The next step is then to connect this with bisections. So when you find a regression, you can then find the commit it was, and automatically tell the cluster. But this is going to need another order of... you know, two orders of magnitude CPU compute! [laughs]

For LibreOffice, from a QA perspective, we also have this bisection. If individual users find a bug and it used to not be there, can they get it down to an individual commit of the user using binaries? They don't have to compile anything. You pull the Git repository down. It's 3GB, 4GB, whatever, and then you can bisect in it, and you can look at versions in between. Of course, once you've found the developer who caused the problem this is much of the problem in programming, it turns out. So then you have someone who feels responsible, and also you've found exactly where it is. That's just been an incredibly powerful tool for getting bugs fixed and keeping quality while we speed up development.

This is the stuff we do for our product in terms of Collabora Office. We do really quite a lot of work there, and lots of hardening. Collabora Office 6 had some things to turn off filters, so you can lock down and manage whole fleets of these things, and say "I don't want this Lotus Pro filter" or whatever, [as] it's had lots of problems. Then, of course, when we put that online, there's an even more heightened sense that it needs to be secure. So what we do is, we put each document inside its own chroot environment, like a container. So the documents can't see each other as they run.

We put almost nothing in that chroot. There's no shell, there are no binaries you can run, really – there's a PDF converter. But there's almost no code, no libraries. If you break out into this thing by getting it to execute an assembler in some weird way, you're not going to see anything. There's nothing there. Except on fonts, and the document, you send them already. You can see your own document. Hooray.

Beyond that, we then have a seccomp [secure computing] filter. We put this Berkeley Packet Filter [BPF] around all your system calls, so the system call interface is shrunk to a tiny amount to, again, stop all of those system calls that once called local privilege vulnerabilities to exploit.

Also, there are almost no device nodes in that chroot. There's a random node. There are very, very few things that you need. But there's no "Here's a view of all your disks, and hey, why not have all your network adaptors, too?" None of that is there. Which is kind of helpful. Again, we're massively constraining that attack surface.

So the expectation is, even if you break out, even if you write a macro that does something stupid – we should be able to enable macros. It's an option, actually, in Collabora Online, that you can have those macros running. Even if you do something stupid on a macro, nothing should happen. As soon as any bad system call happens, we just pull the plug. The whole thing is just torn apart and destroyed. So yeah, it's quite fun. My hope is that it's reasonably secure. Thus far, we haven't seen many security problems there, inside Online.

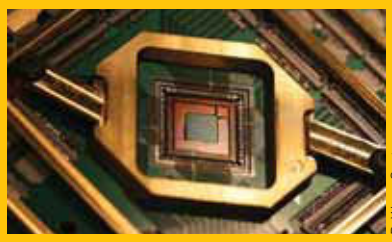
> "If people are paying large QA departments, they should be automating them out, because AI is coming to QA"

QUICK GUIDE

Learning quantum computing in LibreOffice

LibreOffice appears in some surprising places – that's something you learn when you spend time with Michael. While we're queuing up for lunch, Meeks casually mentions that D-Wave Systems, the world's first quantum computing company, uses LibreOffice to run its training programme.

It turns out that teaching people a radical new way to model problems on a quantum computer isn't that easy. Meeks tells us that the company uses a simulator fronted by an XLSM (an Excel Macro-Enabled Workbook file) that dynamically generates elements in the document, and this uses VBA scripts. As LibreOffice now supports VBA, the suite turned out to be the quickest and most efficient way to get the job done.



D-Wave Systems

Above If you want to learn how to use a quantum computer from D-Wave, you'll learn via a simulator that's using LibreOffice

OPINION

The kernel column

As he does every issue for us, Jon Masters summarises the latest happenings in the Linux kernel community

Jon Masters
Jon is a Linux-kernel hacker who has been working on Linux for more than 22 years, since he first attended university at the age of 13. Jon lives in Cambridge, Massachusetts, and works for a large enterprise Linux vendor, where he is driving the creation of standards for energy-efficient ARM-powered servers.

Linus Torvalds has announced Linux 4.18-rc4, noting that things "look pretty normal here, and size-wise this looks good too". Release Candidate 4 (rc4) represents about the mid-way point of a kernel development cycle, which typically lasts a couple of months during which the RCs come weekly. These follow the closing of the 'merge window', a period of time during which disruptive changes are merged that are subsequently soak-tested prior to final release. The 4.18 merge brought in a number of new features, including Mathieu Desnoyers' 'Restartable Sequences' (which we'll cover in more detail next issue) and some additional Spectre security fixes for the Arm architecture.

One important removal in 4.18 is support for the Lustre filesystem. Lustre had been a part of the 'staging' sub-tree maintained by Greg Kroah-Hartman (Greg-KH). The purpose of allowing code in the kernel staging directory is that it be available to those who want to consume not-quite-fully baked new drivers and the like at their own risk. Lustre never quite fell into that category since it has a very active user community and by many accounts is already in widespread production. But the code used in those deployments is based upon an out-of-tree external version of Lustre, not the one in the kernel. After a number of years tolerating this messy situation, the kernel community finally got tired and decreed that Lustre should figure out where it wants to live and go solve that before coming back if it wants to be upstream.

The 4.18 merge window actually closed a day earlier than some might have been expecting. Customarily, new kernel releases come on a Sunday, but Linus had announced 4.18-rc1 from Japan while on a whirlwind tour of Asia. He also noted that he recently hit his 15-year anniversary of working for the Linux Foundation. Time really does fly!

Speculative buffer overflows

Another month, another computer microarchitecture vulnerability with a potential impact upon the Linux kernel. Vladimir Kiriansky and Carl Waldspurger published a paper entitled *Speculative Buffer Overflows: Attacks and Defenses*. The basic assertion is that the industry's previous understanding of Spectre variant 1 (so-called 'bounds check bypass') was incomplete. In fact, a number of sub-variants of Spectre v1 exist, such as the ability to perform speculative buffer overflows.

We have covered Spectre (and Meltdown) in quite some detail before. As a refresher, these vulnerabilities affect computer microarchitecture, which is the implementation of a computer architecture, such as x86 or Arm, in a specific chip family from a given vendor. Spectre v1 in particular affects Speculative Execution, a mode of high-performance microprocessors during which they attempt to guess ahead about which path a program will take following a branch condition, before having all of the information necessary to know for sure that this is the case. Usually, speculation is supposed to function as a 'black box'. If the speculated program flow turns out to be correct, it becomes committed 'architectural state' visible to the programmer (and user), but if it is incorrect, the speculated state is thrown away.

Spectre previously proved that cache side-channel analysis, during which carefully crafted code monitors the behaviour of shared memory resources to time for what has recently been accessed, can be used to infer data that has been loaded as a result of speculation. In the original Spectre v1 exploits, it is observed that a high performance microprocessor might respond to a program bounds check, such as `if (x < 10)`, by speculating that x is less than 10 even if it is not. In the case of the kernel, code might be speculatively run using a user-supplied `x` through a system call that is unsafe beyond a range intended. When combined with a suitable sequence of instructions following the bounds check, it might be possible to leak sensitive kernel data to an attacker.

We previously mitigated this in the kernel by using tools such as Dan Carpenter's Smatch to look for these vulnerable code sequences, and inserted 'clamping' masking macros following the bounds check that will force the value speculated to lie within the range of the bounds check.

The latest Spectre v1 twist acknowledges that following such a bounds check, there might be stores (writes) to memory that could be abused under speculation similar to a conventional buffer overflow – except in this case, the buffer overflow only happens speculatively, as does the overwrite of data structures following the buffer. While all of this will be thrown out once speculation is unwound, there is a time

window during which kernel-code control flow could be speculatively resteered based upon this overflow and cause data leakage.

As a result of Vlad's work, which apparently won a significant bounty prize for its discovery, the existing mitigations are being updated to account for the sub-variant, adding additional uses of kernel macros such as array_index_nospec that perform a clamp on the upper bound. You can read Vlad's paper at **https://people.csail.mit.edu/vlk/spectre11.pdf**.

The Linux Kernel Mailing List (LKML) finally has an official archive. For many years, various individuals and groups maintained copies of the list, including gmane.org, marc, and others, but there was never something maintained on the official kernel.org infrastructure. That has now changed with the unveiling of **lore.kernel.org**, which features an online web-accessible archive as well as Git-clonable copies, and a news-compatible reader at **nntp.lore.kernel.org** for us old-school types.

Matthew "Willy" Wilcox posted yet another iteration of his 'Convert page cache to XArray' patch series. As mentioned previously, XArray is an attempt to replace the traditional implementation of radix tree data structures in Linux with a structure that scales better and is designed with the actual real-world uses of radix tree in mind. Matthew is requesting inclusion of XArray in Stephen Rothwell's 'linux-next' kernel soak-testing tree ahead of an effort to merge in 4.19.

> "Linus noted that he recently hit his 15-year anniversary of working for the Linux Foundation"

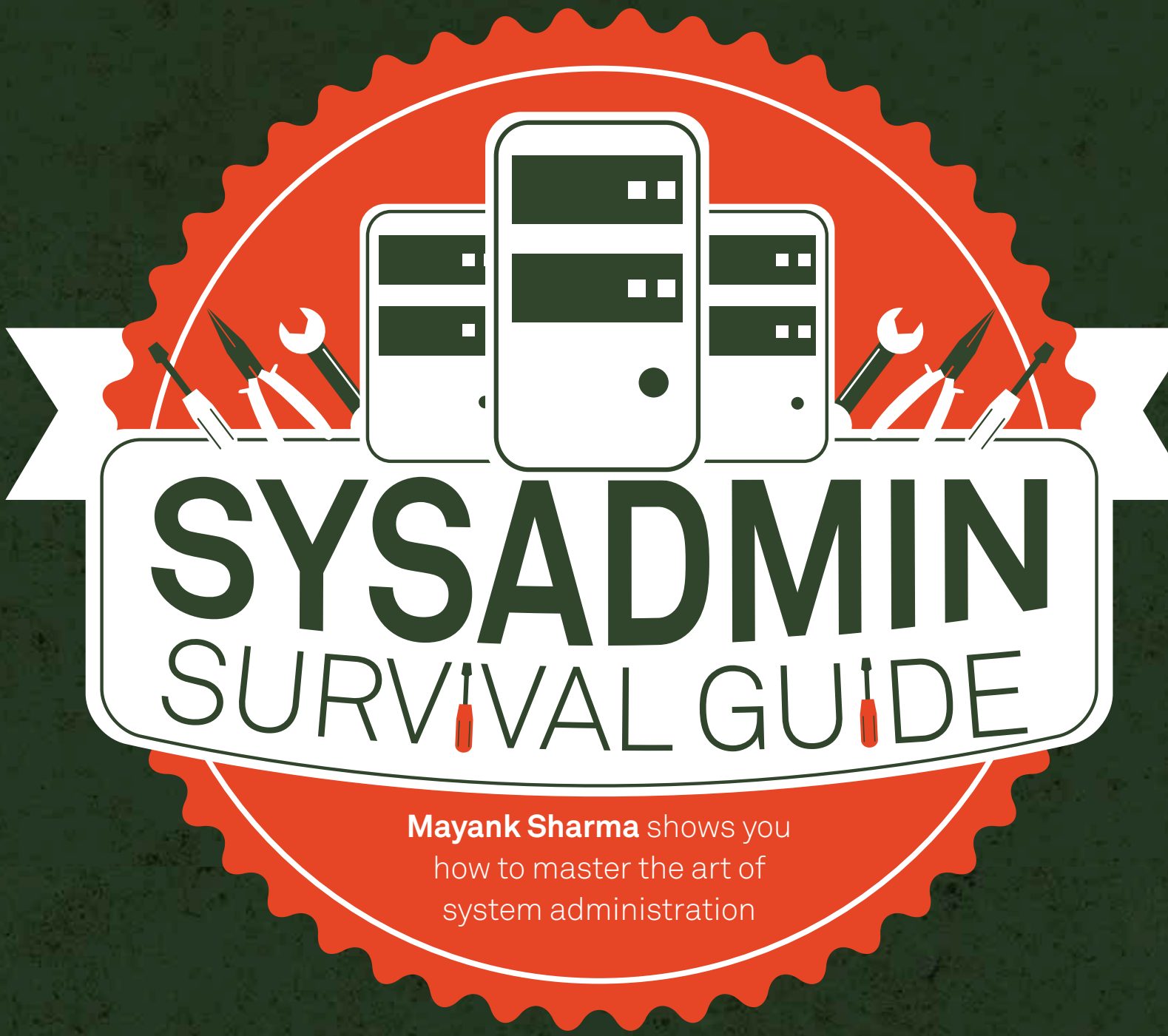
Chang S. Bae (Intel) posted an updated patch including "infrastructure to enable FSGSBASE". The new FSGSBASE 64-bit instruction set extension in x86 allows the x86 segmentation registers (an archaic anacronism that still exist in some limited form on 64-bit x86) to be updated from less privileged levels. This means that the kernel entry code can be modified to save a SWAPGS instruction on every system call, reducing overhead.

Yu-cheng Yu (Intel) posted four different patch sets laying the groundwork to support Intel's upcoming Control Flow Enforcement (CET) technology. CET is a "processor family feature that prevents return/jmp-oriented programming [ROP] attacks". It's also fundamentally incompatible with some of the Spectre-v2 mitigations such as retpoline, so the rollout is being handled with some care. Nonetheless, CET is a great technology and we will cover its upstreaming progress in more detail in a future column.

Baolin Wang (Linaro) posted a patch series entitled "Add persistent clock support", which lead to a spirited discussion about how Linux handles timekeeping sources across architectures and power states. The generic concept of a timekeeping source is represented in Linux using the 'clocksource' framework. This is how your laptop keeps its view of time in sync with the RTC (Real Time Clock) chip, which is used to persist timekeeping across reboots and suspend/resume cycles. In some cases, a hardware platform may need to use a bespoke combination of time sources that differ between runtime and suspend time. Boalin wanted to propose a new framework ('persistent clocks') to do this, but Thomas Gleixner shot this down quite passionately, dismissing the patches and suggesting that we should "look at the problem itself". He noted that when multiple clocksources are in use, the existing framework can be enhanced rather than adding a new framework. Boalin said he would change course.

Michael Kelley (Microsoft) posted patches entitled "Enable Linux guests on Hyper-V on ARM64" which "enables Linux guests running on Hyper-V on ARM64 hardware". The patches plug into all of the existing Hyper-V frameworks, adding new hypervisor 'hypercalls' that are Arm-specific, while updating other previously x86-only code to make it cross-architecture.

Andrey Ponomarenko announced that "a new project has been created to collect the list of computer hardware devices with poor Linux compatibility based on the Linux-Hardware.org data". This has been tried before; the latest effort can be found at **https://github.com/linuxhw**.



SYSADMIN SURVIVAL GUIDE

Mayank Sharma shows you how to master the art of system administration

AT A GLANCE

- Storage & disaster recovery p20
  Manage storage devices and develop a good backup strategy.
- Create a NAS p20
  Use OpenMediaVault to create a network-accessible storage pool.
- Network essentials p22
  Shield the network with a firewall and learn how to shape traffic.
- Scripting & automation p24
  Use Kickstart to automatically provision computers on the network.
- Administer with Fabric p24
  Make changes to network machines with this helpful Python library.
- Cloud & virtualisation p26
  Get to grips with the cloud and learn to provision virtual machines with Vagrant.
- Manage KVM p26
  Use Virtual Machine Manager to easily deploy KVM-based VMs.
- Logging & monitoring p28
  Keep an eye on your realm using the osquery and logwatch utilities.

The field of Linux system administration has changed tremendously over the last few years. Be it the exodus to cloud computing or the proliferation of virtualised components and containerised apps, modern IT infrastructure has gone through a paradigm shift in a short span of time. No surprise, then, that organisations are on the look-out for administrators who are equipped with the necessary skills to handle this shift in enterprise technology.

Matt Simmons, Linux Systems Administrator for a leading American aerospace provider, believes system administration can be classified into three broad categories. What he calls traditional system admin is "where you only program to automate small things on a per-machine basis. The kind of role where you only write quick shell scripts to back things up, and maybe to grab some monitoring data that you throw into Nagios, Icinga, or something like Cacti. Your machines are probably mostly physical." He adds that admins of such networks spend most of their time on managing resources per machine and a sizeable chunk of their budget goes towards procuring hardware or updating them: "This is the minority, and it is shrinking."

The other end of the spectrum is all cloud: "You have automated jobs creating images regularly according to controlled rulesets, so that when you need to create a machine, it happens without any doubt as to what is on it and what it is doing, and the machine shuts down as soon as is possible; maybe because that service isn't needed any more, or maybe because the next image is ready to take over for it," says Matt. He adds that admins of such deployments spend their time looking for ways to optimise spending and consolidate and eliminate provided services. According to Matt this kind of administration is the minority as well, but is growing rapidly.

> "Today rarely do people have to get acquainted with boot loaders, kernels, file systems and device drivers, which limits them"

The majority that seems to be holding steady, in Matt's opinion, is a meeting of the two. He says the role in this segment has "probably shifted from 'systems administration' to what might be considered 'IT Operations'." Admins here use configuration management, and will surely have some services in the cloud: "You write automation whenever you get the chance. You probably work almost exclusively from tickets, you have CI/CD workflows for at least some of your code, and if you have a team, you probably do some sort of team programming and code review. Your machines are almost all virtual, and provisioned automatically without much, if any, interaction from you."

Abhas Abhinav, founder and hacker-in-charge of DeepRoot Linux who supports over 250 servers for clients all over India, says that as system admin is getting easier, the level of commitment and respect for the "craft of system admin" isn't what it once used to be. "On the other hand, [because of] the fact that systems are now easier to provision or procure because of the ubiquitous nature of VMs, many system admin skills are not getting enough attention. Today rarely do people have to get acquainted with boot loaders, kernels, filesystems and device drivers, and this limits the sort of problems they can manage and solve."

If you are a new system admin, this feature shows you what you need to know to get started. For the more experienced, it equips you with knowledge of the tech you need to be familiar with to stay relevant.

Storage & recovery

A good system admin uses storage systems efficiently and also ensures hardware failures don't bring down the house

Data is one of the most critical parts of any business, which means managing it is one of the core responsibilities of any system administator. The key to a good storage management policy is that it should strike the right balance between keeping the data safe yet accessible.

Before you can design a storage policy for an organisation, you need to understand how Linux deals with storage and familiarise yourself with the tools at your disposal. All it takes is a couple of clicks to add storage to a cloud server; provision a drive and then attach the disk to an existing VM. In the real world however, you'll first to decide on the type of device (e.g. hard disk, SSD) along with the interface to connect it to your server (e.g. USB, SATA, PCIe). When the Linux kernel detects a device, it tells udev, which then creates a representation of that device in the `/dev` directory. These device files are the way the kernel provides access to devices for applications and services. The storage devices are called block devices, and a good practice is to refer to them using their Universally Unique Identifiers or UUIDs. The next piece of the puzzle is partitioning – or rather its more dexterous cousin, logical volume management (LVM). While the traditional partitioning schemes MBR and GPT are much simpler, they are rather inflexible. LVM, on the other hand, combines one or more devices into a single logical volume group. You can then dynamically create, resize and delete volumes in a volume group to reallocate space. It also offers other benefits such as snapshot management.

> "Mastering rsync is one of those evergreen skills that'll never go out of vogue"

Linux systems give you several options for partitioning, with `parted` and its graphical frontend `gparted` leading the pack. To manage volumes you need to master `lvm`.

While you can (and should) supplement all storage mechanisms with backups, it's always a good idea to use a RAID system to distribute data across multiple disks. With the plummeting cost of storage, a RAID setup helps avoid data loss and also minimises the downtime associated with hardware failures. While RAID can be implemented by dedicated hardware, it can also be implemented by software. Use the `mdadm` command to build and use software RAIDs. No matter how you implement it, make sure you are well versed in the different RAID levels to pick the one that works best for your organisation.

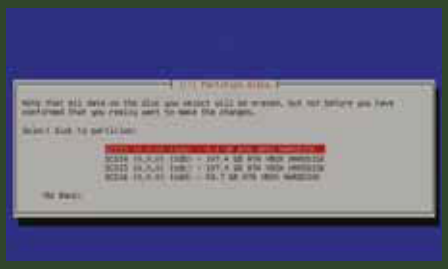
Another popular storage deployment skill that should be in the repertoire of an admin is the ability to set up a network-attached storage (NAS) device. OpenMediaVault (**www.openmediavault.org**) can make this process effortless. It supports all the popular deployment mechanisms, including software RAID, and can be accessed using

EXPERT TIP

RAID is not backup

Don't use Mirrored RAID as an excuse to avoid taking backups. You don't want to be in a situation where a fried machine renders both disks useless.

HOW TO
Create a NAS with OpenMediaVault

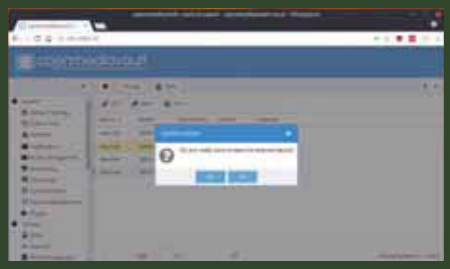


1 Install OMV

Installing OMV is pretty straightforward. Download the ISO, transfer it to a USB disk and run through the basic steps. Since you have multiple disks connected to the NAS box, make sure you select the right installation target during the partitioning step.

2 The web interface

When you're done, reboot the computer, which will drop you to the login shell. Don't log in but instead head to the web address displayed on the console. This brings up OMV's web interface, from which you can manage all aspects of the NAS server.



3 Set up storage

You'll first need to format the drives before you can use them. Head to Storage > Disks to view all the attached disks. Select the drive and click the 'Wipe' button. After you've erased a drive, head to Storage > File Systems to create a file system on the drive.

Below Grsync is a graphical front-end to rsync that exposes virtually all of rsync's command-line options

all the leading network protocols (see the walkthrough below).

Before you can press the disk into active service, it needs to have a file system on it. Most Linux distributions default to the EXT4 journalled filesystem, but there are several others that you should be familiar with. There's NTFS, which is useful for interoperability with Windows machines, and XFS, which works well for housing database files. The needs of organisations are, however, much better satisfied with next-generation file systems, such as ZFS and Btrfs, that perform better than traditional file systems and are more serious about data integrity.

Once you've got a handle on the storage systems in the organisation, it's time to work out a plan for backing up all the data. Irrespective of the technology that powers it you'll first have to spend some time answering questions like:

- What data is to be backed up?
- Where will backup data be stored?
- How often will backups be performed?
- How long will backups be retained?
- How will backups be accessed or restored?
- What system or technology will perform the backups/restoration?

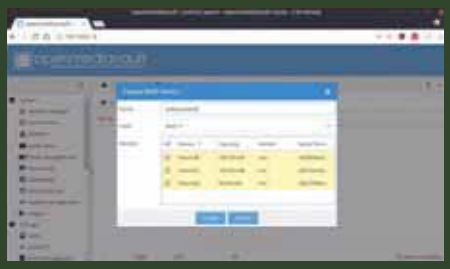
The answers to these (and other) questions depend on various factors such as cost and the amount of acceptable downtime. For example, the availability advantage of remote cloud storage diminishes with an increase in the amount of backed-up data.

There are a plethora of reputable open source backup tools and a number are designed specifically to handle large amounts of data. One of the industry favourites is rsync, which scales well and offers enough dexterity to power desktop backup apps as well as enterprise ones like BackupPC. Mastering rsync is one of those skills that'll never go out of vogue.

QUICK TIP

rsync and BackupPC

rsync is one of the handiest tool to copy and sync files across the network. But if writing scripts isn't your forte you can deploy and master BackupPC, which is a backup app written for the enterprise and uses rsync in the background.



4 **Use RAID**

Instead of using the disks individually, OMV can tie them into a software RAID. Head to Storage > RAID Management and select the disks you want to use in the RAID, as well as the RAID level. Wait for the RAID to initialise before creating a file system.

5 **Users and shares**

Head to Access Rights Management > User to add or import users. Next, you'll have to add shared folders from the Access Rights Management > Shared Folders menu. Use the pull-down menu to select the volume for the folder.

6 **Enable shares**

Finally, enable a network service – which users will use to access the shared folders – under the Services section, and register the shared folders with the service under the Shares tab. Your users can now access these shared folders across your network.

Network essentials

Networks are a crucial building block of a modern day organisation, and managing them well is an essential skill

Although administrators interact with real-world network hardware less frequently than they once did, familiarity with traditional networking still remains a crucial skill. In addition to grasping the underlying technology and the protocols at play, you must know how to configure your network peripherals and how to troubleshoot the connections. You should be familiar with the process of configuring networking in Linux and also have knowledge of the configuration files involved.

The nature and composition of your organisation will heavily influence how you set up your network. A business that is starting out often has only one main server that pretty much provides all the networking infrastructure: a DNS, DHCP, file, mail and web server all rolled into one. But very few larger businesses would trust their entire IT infrastructure to an individual host. Most admins would avoid such a single point of failure, but a small business can't afford to have a host for the individual service.

Irrespective of the size of the network you manage, the one networking skill that you must grasp firmly is tuning the firewall. The Linux kernel uses Netfilter to facilitate key network process such as Network Address Translation (NAT), packet filtering and packet mangling. The `iptables` command is the user-space management tool for Netfilter. Many distributions have graphical tools for configuring the Linux firewall and the excellent Shorewall utility (**http://shorewall.org**) makes it easy to configure the firewall.

Above You can use server distributions such as ClearOS to deploy network services with ease

> "The one networking skill that you must grasp firmly is tuning the firewall"

Still, do yourself a favour and spend time tinkering with the `iptables` command to really appreciate its usefulness.

Another aspect of the modern workspace is that it extends beyond the confines of the physical office. If your users are not located locally, you need some way of connecting them as if they were local by setting up a virtual private network (VPN). A VPN is, in essence, a private network that runs over a public network, and one of the best tools to help you get the job done is OpenVPN. The package is available in the official repositories of all mainstream Linux server distributions. Once you have it up and running, spend some time configuring it to securely expose the resources in your organisation's network to other users over a public network.

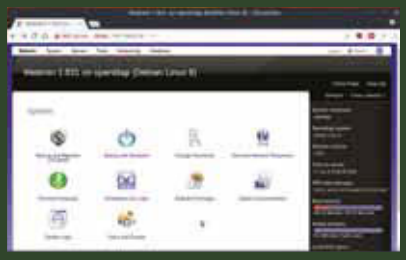
While it's always a good idea to configure and roll out these key network infrastructure components manually, there are several specialist distributions such as ClearOS (**www.clearos.com**) and NethServer (**www.nethserver.org**) that are designed to deploy many of these common services.

Shape traffic

Netfilter is a complex firewall application that can regulate network traffic and also shape it. Shaping network traffic or packets involves altering the stream of packets passing through the network. Admins usually use it to regulate the bandwidth for certain types of connections. A simpler utility to

QUICK GUIDE
Master Webmin

Webmin (**www.webmin.com**) is a configuration tool that can be used to control all aspects of your remote server, such as setting up a cron job, reading logs and managing running processes. Using it you can dispense your admin duties from a web interface. Instead of manually editing configuration files and fiddling with command line switches, Webmin helps you configure different aspects of your system, which then automatically update the relevant underlying config files. Webmin can manage network services as well as the host system. For instance, you can use the tool's interface to create and configure virtual hosts for the Apache Web server, and set up a Samba file-sharing server just as easily as you can create and manage user accounts and set up disk quotas. Webmin's intuitive dashboard contains a list of modules each of which is responsible for managing some service or server, such as the Apache web server, the firewall, software packages and so on. When installed, Webmin reads config files for all servers and services on your system from their standard install locations.



Above Webmin saves you the trouble of having to memorise numerous parameters

shape bandwidth is trickle, with which you can manage the upload and download speeds of applications in order to prevent one from hogging all the bandwidth.

Trickle regulates speeds by delaying the data transferred over a socket. Trickle provides an alternate version of the BSD socket API, with the effect that socket calls are now handled by trickle. Speed is limited by controlling the amount of the data written to or read from a socket.

You can run trickle in either the standalone mode for shaping individual downloads, or configure the trickle daemon to set global limitations and priorities. For example, the following command

```
$ trickle -s -d 30 ncftpget ftp://ftp.foo.bar/Linux/somedistro.iso
```

invokes trickle in standalone mode (`-s`) and sets the download rate to 30KB/s for downloading the specified ISO image file, using the `ncftp` non-interactive FTP download client. You can use the trickle daemon to set global limitations and priorities. When you invoke trickle without the `-s` switch it checks to see if the trickled daemon is running, reads the `/etc/trickled.conf` file and applies any rules it finds listed there. By default the file has an example entry that'll make sense with a cursory look at trickle's man page. Browsing through the file, you'll realise that trickle can do a lot more than just set upload and download speeds. It can also define a per-application priority, along with time- and length-smoothing parameters. Time-smoothing enables you to define the time interval for the application to transfer data. Large values will produce bursts in sending and receiving data, while smaller values ensure a smooth and continuous data transfer.

> "Shaping network traffic or packets involves altering the stream of packets"

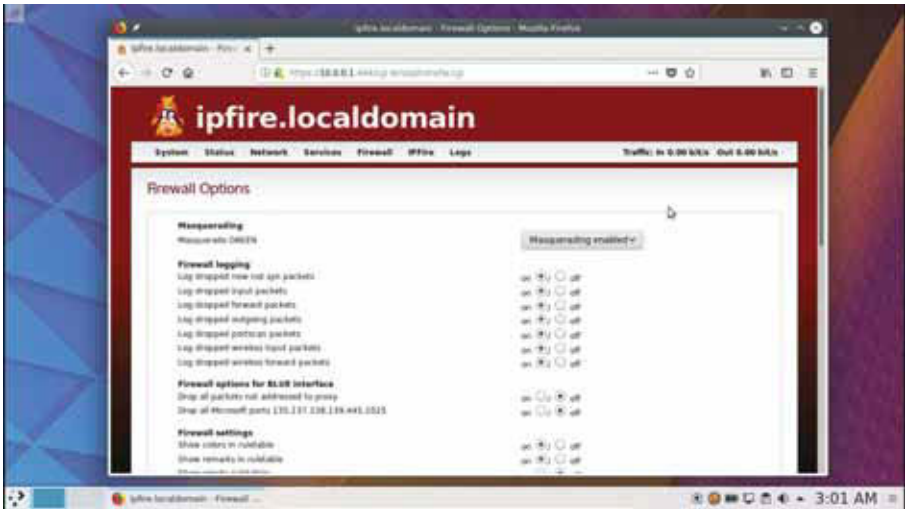
Trickle does have some limitations. For one it can only work on TCP connections, so you can't use it to regulate a UDP stream connection such as DNS. Furthermore, since it uses dynamic linking and loading, it can only work with applications that use dynamic libraries (Glibc). Statically linked applications are thus not compatible with trickle. To determine if you can use trickle with an app, use the `ldd` command, which gives you a list of all shared libraries. For example:

```
$ ldd /usr/bin/ncftp | grep libc.so
```

EXPERT TIP

No slashing

When using rsync, don't put a trailing slash at the end of the source directory. This copies the directory along with its contents, instead of scattering the files inside the backup directory.



Above Instead of configuring iptables manually, you can use a specially tuned firewall distribution such as IPFire (pictured above, **www.ipfire.org**) or the graphical Shorewall utility

PRODUCTS

Key infrastructure tech

Learn to administer these four useful server apps

1 OpenLDAP

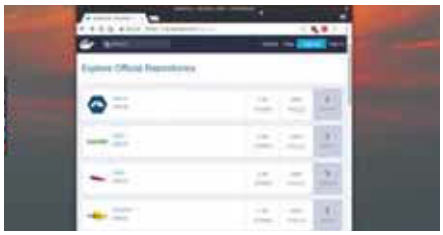
www.openldap.org

All companies need a directory server and OpenLDAP is the popular open source implementation of the Lightweight Directory Access Protocol (LDAP). Despite being lightweight, it's a fully featured suite of apps and development tools.

2 Kolab

www.kolab.org

Kolab is an open source, unified communications, collaboration and groupware suite. It's built with security, privacy and user control in mind and provides solutions for an email server, directory service, web service, Calendar, Tasks, address books and more.



3 Docker

www.docker.com

The shiny new and popular piece of tech for managing and running apps in virtual containers is also a boon for admins. You can use it to deploy conflict-free and production-like development environments for your users.

4 MariaDB

https://mariadb.org

Developed by the original authors of MySQL, MariaDB is one of the most popular database servers in the world. Many companies use it directly or via apps built around it, so it pays to make sure you are familiar with it.

Automation

Smart admins are lazy – they automate mundane repetitive tasks using scripts

As a savvy admin you should try to automate all your regular tasks, such as flushing caches and removing temp files, locking out idle accounts, backups and so on, using scripts. Scripts standardise these administrative chores and free up admins' time for more pressing tasks. You should spend time hashing out a plan to perform repetitive tasks with as little involvement or intervention as possible.

Bash and Python are two of the most popular scripting languages for automating system administration tasks. Generally speaking, lengthier scripts for important tasks such as backups and user resets are executed on a schedule. To that end, spend some time getting familiar with the time-based job scheduler, cron.

> "The best work is work you don't have to do. Eliminating work isn't lazy, it's smart"

Another task that you must automate is the installation and maintenance of the computers in your network. Manual installs don't scale and are prone to errors, especially when done repeatedly. Fully Automatic Installation (FAI, **https://fai-project.org**) is a set of Perl scripts that enables you to run an unattended Debian install. Similarly, Kickstart helps you automate the installation of RPM-based systems such as CentOS and Fedora.

EXPERT TIP

Matt Simmons on scripting

The best work is work that you don't have to do. Eliminating work isn't lazy, it's smart. Cut extraneous work and concentrate on only that which needs to be done.

Kickstart can install not only the operating system but also all the applications you expect to run.

To make everything run as quickly as possible, you can store Kickstart files and Linux packages on a local Apache web server, which then serves as your network installation server. To build your installation server, start by installing a barebones CentOS system. Once the hardware and operating system are set up, make sure the server has a fixed IP address, say 192.168.1.100. Then install the Apache web server from the distribution's repositories. Next, copy over the installation files from the CentOS DVDs to this server with something like `rsync -arv /media/CentOS_7 /install`.

Ideally you should set up the `/install` partition on a separate disk or a logical volume for more flexibility. If you create the install directory outside Apache's document root directory (`/var/www/html/`), you must create a configuration file to point the web server to the correct directory, such as:

```
# nano /etc/httpd/conf.d/install.conf
Alias /install/ /install/
<directory /install>
        Options Indexes
        AllowOverride None
        Order allow,deny
        Allow from all
</directory>
```

Before you can access your install server you'll have to make sure the firewall on the machines in your network allow HTTP traffic over port 80 using `iptables -I INPUT -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT`.

Other machines on your network will now be able to connect to the installation server. Next you need a kickstart file to automate the installation. The most convenient way to create one is to use the graphical Kickstart Configurator tool from the repositories. Besides the package selection, you need to pay attention to two

PRODUCTS

Automate admin tasks

Conduct admin business over several machines with Fabric's Python library

```
from fabric.api import run
def uptime():
        run ('uptime')
```

1 Define a function

Enter these lines in a file called `fabfile.py` and then call the function with `fab uptime`. You can also run functions on multiple remote servers with `fab -H localhost,192.168.0.100`

2 Change context

Fabric's context managers are used with Python's `with` statement. You can use the settings context manager if you temporarily need to run a command as a different user, such as `with settings (sudo_user='mysql'):`.

```
from fabric.api import *
env.hosts = ['192.168.0.100','192.168.0.101']
def upgrade_all ():
        sudo ("apt update")
        sudo ("apt -y upgrade")
```

3 Update several machines

The env.hosts variable lists all computers in your network. When you call the function with `fab upgrade_all` it'll connect to the remote machines, refresh their repos and install any updates.

```
from datetime import datetime, date, time
now = datetime.now()
def fetch_logs():
        get (remote_path="/tmp/logs.tar.gz", local_path="/logs/{}_log.tar.gz".format(now))
```

4 Download remote logs

The `get` command is useful for downloading logs from a remote system. This example saves them to the local PC with a timestamped file name.

important things in the kickstart files to work with an installation server: the installation method and the network setup.

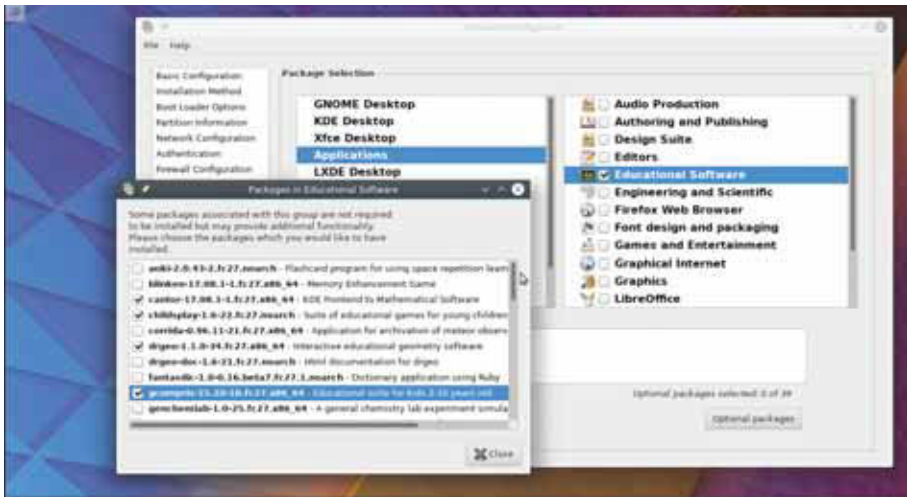
The clients we'll be setting up should fetch packages from the network server we've just set up, instead of from the CentOS installation DVD. To change the installation medium, head to the Installation Method section in the Kickstart Configurator, toggle the HTTP setting and enter the location of the installation server – **http://192.168.1.100/install/** – in the adjacent textbox. Then jump to the Network Configuration section and make sure that the network device you add uses the DHCP server.

Place the generated kickstart files under the web server's document root directory, in a directory such as `/var/www/html/kickstarts`. To install a client using the kickstart files, boot from the minimal boot CD, and at the boot screen point to the kickstart file you want to use, such as `linux ks=http://192.168.1.100/kickstarts/gnome-workstation.cfg`. Once you have everything set up, rolling out new machines

> "The clients we'll be setting up should fetch packages from the network server"

becomes just a matter of pointing to the correct kickstart file. You can set up new machines, reset old ones for new employees, or change a web server into a mail server with a single command.

While the installation is automated, you still have to initiate it manually. The key to fully automating the process is the Pre-Execution Environment (PXE). A PXE server acts as the network boot server and broadcasts a DHCP 'discover' request that includes the name of the boot server and the boot file. The PXE-enabled remote client responds to the request, downloads the boot file and then executes it to begin the installation, which itself has already been automated by you.



Above Generate as many kickstart files as you want, each with different package selections

Configuration tools

That takes care of the deployment. But what if you have to make a configuration change in a deployed machine, such as the firewall change we listed earlier to allow incoming HTTP traffic? There are several tools including Puppet, Chef, Foreman, SpaceWalk, Ansible and Fabric that help you orchestrate various configuration operations.

The advantages of Fabric (**www.fabfile.org**) and Ansible (**www.ansible.com**) is that unlike the other solutions these two have no server daemons and are agent-less. In essence they're really just a set of commands that you install on any system from which you wish to manage clients. Both Fabric and Ansible rely on SSH to conduct their business, and you can install them using your distribution's package manager.

One difference between Fabric and Ansible is that you can get started with Fabric without much delay, while with Ansible you'll have to spend some time setting it up. However, many experienced admins consider Ansible to be ideal for large and complex networks since it can better model multi-tier infrastructure. On the other hand, Fabric uses Python for authoring which is much simpler to understand ▸

EXPERT TIP

Abhas Abhinav, DeepRoot Linux

Automate everything that you would need to do more than once. After five times, generalise it. And then, share it under a free-software licence.

PRODUCTS

Run automated installs

Save time and effort by automatically provisioning machines using these solutions

1 MAAS
www.ubuntu.com/server/maas
Canonical's MAAS is easy to set up and can provision Ubuntu, CentOS, RHEL and SUSE servers. It was designed to compliment Canonical's service orchestration framework called Juju that enables you to easily deploy services with its charms architecture.

2 Foreman
www.theforeman.org
Foreman's a popular choice for provisioning machines and can deploy lots of mainstream distributions. One of the reasons for Foreman's popularity is interoperability; it can work with a host of popular application provisioning tools such as Puppet and Chef.

3 Fog
https://fogproject.org
With Fog you can image an installation and deploy it to other computers on the network. It also makes light of regular administration tasks such as installing software, and can manage large networks that may be spread over multiple locations.

Cloud & virtualisation

Learn and master the new ubiquitous tech that has completely revamped modern IT infrastructure

Perhaps the one biggest reason for the move towards computing in the cloud is convenience. You get technically superior access without the overhead of maintenance and upkeep. Cloud computing offers several other advantages to the modern IT infrastructure, and like any other resource on the network, it has to be looked after by the system administrator.

There are various models of cloud services. Of these, Infrastructure-as-a-Service (IaaS) is the most pertinent to system administrators. IaaS enables businesses to purchase resources on demand, as and when required, instead of having to buy hardware outright. These are typically delivered in the form of virtual private servers (VPS) that are made up of virtualised peripherals. You should have the skills to evaluate IaaS providers and select the one that best meets the requirements of your organisation.

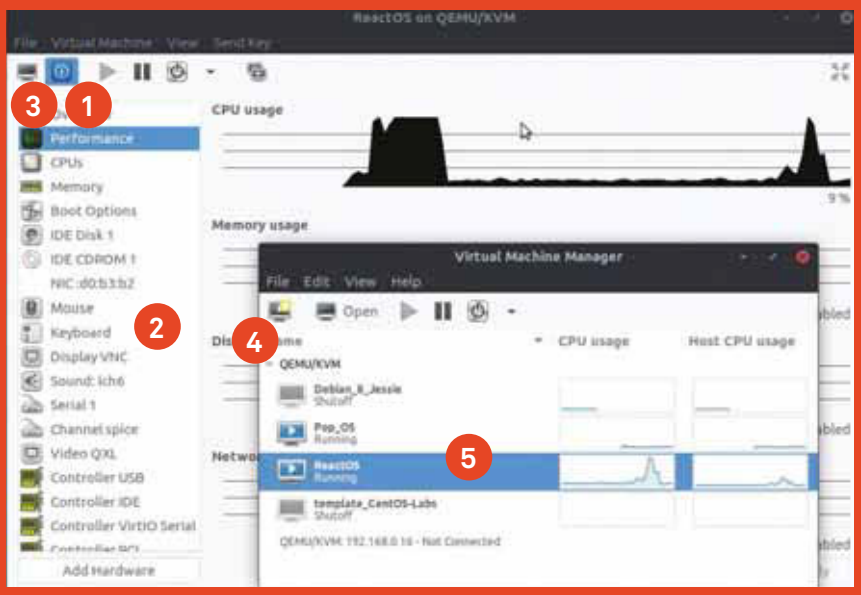
Virtually all cloud vendors have a web-based graphical interface for interacting with the virtualised resources. The good ones even have APIs that enable you to access all their functionality via scripts for automation.

> "VirtualBox's VBoxManage CLI can be used to automate essential VM tasks"

The best way to get to grips with this new dimension of the IT infrastructure is to sign up for an account and leaf through their APIs to understand how to integrate them with your scripts and other automation tasks.

KVM, or Kernel-based Virtual Machine, has been the default hypervisor on Linux since 2007. It is a set of kernel modules that when loaded converts a Linux server into a hypervisor. KVM depends on libvirt, which provides a convenient way to manage VMs and other virtualisation functionality, such as storage and network interface management. You can only create KVM-based VMs on servers that have processors with hardware virtualisation extensions, either AMD-V or Intel VT-x. VirtualBox (**www.virtualbox.org**) is another hypervisor that you'll need to familiarise yourself with for the sake of end users on your network. The good thing about VirtualBox is its VBoxManage CLI that you can use to automate essential VM tasks such as snapshots.

The more useful virtualisation tech you must get to grips with is Linux containers, especially Docker (**www.docker.com**). Docker enables you to bundle any Linux app, with all its dependencies, into its own environment.



QUICK GUIDE

Manage KVM with virt-manager

The Virtual Machine Manager is the open source graphical front-end for creating KVM-based VMs, and is available as `virt-manager` in the repositories of most distributions. The app uses the qemu-kvm hypervisor, which is a version of the QEMU machine emulator modified by the KVM developers.

The virt-manager app is written in Python and is very intuitive to operate. You can use the app to easily create new VMs, monitor them and make configuration changes. virt-manager also includes a VNC and SPICE client that displays a full graphical console to the running VM.

1 **VM details** Click this button to view details about the virtual hardware attached to the VM

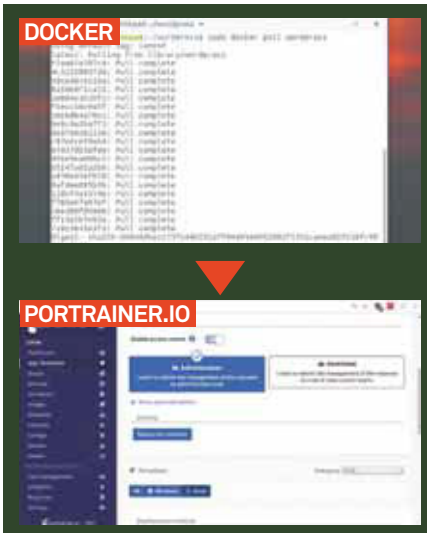
2 **Virtual Hardware** You can configure virtual peripherals or add new ones

3 **Graphical console** Use this to peek inside a running VM via VNC or SPICE

4 **Create VM:** This button brings up a five-step wizard to create new VMs

5 **List of VMs:** virt-manager's main dashboard displays all added VMs running on local and remote hypervisors

Above You can grab a host of Debian-based appliances that have been optimised for virtual machines and the cloud from **www.turnkeylinux.org**



QUICK TIP

Docker and Portainer.io

Using Docker via the terminal isn't all that cumbersome and the tool is well documented. To make your life easier you can use Portrainer.io (**https://portainer.io**), which is an open source, web-based graphical front-end that supports all features exposed by the Docker API.

You can then run multiple instances of the containerised app, each as a completely isolated and separated process, with near-native runtime performance.

Provision virtual

Next you should familiarise yourself with Vagrant (**www.vagrantup.com**), which helps you make consistent virtual environments available to your users with a few keystrokes. Vagrant supports all major virtual platforms such as VirtualBox and VMWare, and plays nicely with all the well-known software configuration tools such as Chef, Puppet, Ansible, Fabric and more.

To get started with Vagrant, fetch and install the latest binary from its website. You'll now have to create a Vagrant configuration file that defines all the characteristics of the VM from a template. You can search for templates based on various operating systems and pre-defined purposes from **https://app.vagrantup.com/boxes/search**. So for example, `vagrant init centos/7` creates a VM based on the CentOS 7.5.1804 distro.

The command creates a file called Vagrantfile under the current directory, and is the main configuration file that defines all the attributes of the VM. If you want to make changes to a VM, you'll need to edit it. You'll need to be well-versed with the anatomy of a Vagrantfile in order to modify or create one as per your needs. For now just use `vagrant up` to create an actual VM from this file. This tells Vagrant to create a new VirtualBox machine based on the base image specified in the Vagrantfile. It'll do so by copying the virtual hard disk files from the remote server.

> "You'll need to be well-versed with the anatomy of a Vagrantfile to edit it"

Once it's done, you'll have a fully featured CentOS 7 VM running headless in the background. Use the `vagrant ssh` command to connect to the machine via `ssh`. You can now interact with the VM like any other CentOS installation. When you're done, type `exit` to drop back to your host and use `vagrant halt` to turn off the VM.

You can now modify the Vagrantfile to customise the installation. If you are running a web server inside the VM, you'll want to forward port 80 by adding this line:

```
config.vm.network "forwarded_port", guest: 80, host: 8080
```

You'll now be able to access port 80 on the guest via port 8080 on the host. Type `vagrant up` to start the VM once again. You can also make changes to the Vagrantfile while the VM is running; in that case, type `vagrant reload` to load the VM with the modified settings. Similarly, you can put all the operations you might do with a freshly minted CentOS box – such as installing some apps – inside a Bash shell script and then point to it from the Vagrantfile with:

```
config.vm.provision "shell", path: "post-install.sh"
```

Now bring up or reload the VM. Thanks to this line, Vagrant will execute the `post-install.sh` script after the VM is up and running. There's a lot more you can do with Vagrant, though, so make sure you read through its documentation. ▸

Logging & monitoring

Be thorough and take steps to ensure all the IT infrastructure in your realm keeps chugging along smoothly

Everything from databases to daemons keep logs of their activity and managing these is as important a function as deploying servers. It's the responsibility of system administrators to gather actionable bits of information from this deluge of messages.

Log data usually ends up in the `/var/log` directory in a variety of files usually placed by the syslog daemon. Syslog has been the standard Unix format for several years, but there have been a couple of replacements, with rsyslog being the most recent and popular one. Make sure you are fully versed with the configuration of the logging daemon on your servers.

Another aspect of managing logs is to control their size and archive duration, which for some types files such as access logs is mandated by regulatory policies. To help you keep the logs manageable you need to rotate them, which involves flushing the old ones to an archive file. The logrotate utility is a popular choice for this task as it implements a variety of log management policies.

A good sysad should be fully aware of the limited usefulness of this data, in terms of time, and must automate the process of filtering and summarising it using tools such as Logwatch (see right). Furthermore, it's important to note that the `syslogd` daemon is a passive tool that awaits inputs from apps and doesn't go out and actively query them. For this you need a monitoring tool

> "Syslog has been the standard Unix format for several years"

like Nagios (**www.nagios.org**), one of the most popular and extensively used network monitoring tools, which you can use to streamline the management and monitoring of your network. You can use it monitor nearly all devices and services that have an address and that can be contacted via TCP/IP. ■

QUICK GUIDE

Analyse with Logwatch

One of the least interesting part of an admin's job is to scroll through logs in order to spot potential issues. This is where log filtering tools like Logwatch come in. It parses, analyses and filters logs, and then generates daily reports on your system's log activity. The utility is available in the official repositories of all popular server distributions, including CentOS and Debian.

Above Logwatch reports are categorised by services; you can control the level of verbosity

HOW TO

Use osquery

```
SELECT * from logged_in_users;
SELECT * from last;
```

1 Check on users
You can use osquery to keep an eye on the users logged into your servers with the first command shown above, while the second shows the list of previous logins. Look for logins from unknown IP addresses, and even more so if there are multiple users logging in from an unfamiliar host, which should be a red flag.

```
SELECT * from iptables;
SELECT * from listening_ports;
```

2 Check firewall
These queries help keep an eye on the firewall. If the first one doesn't produce any output it means there's no firewall – which isn't a good thing, especially on a server. The second lists all the listening ports and will help you find back-doors on the server.

```
SELECT * from kernel_info;
SELECT name, size, used_by, status from kernel_modules where status="Live" order by size;
```

3 Check kernel
You'll also want to run these queries periodically and compare their output against older results for any changes. The first retrieves information about the current kernel to help identify outdated kernels, while the second lists all loaded kernel modules.

```
SELECT pid, name, uid, resident_size from processes order by resident_size desc limit 10;
SELECT name, path, pid FROM processes WHERE on_disk = 0;
```

4 Watch processes
The first query displays the 10 largest processes arranged by size, while the second displays processes that don't have a binary associated with them. You should immediately terminate that process.

ON SALE NOW!

Available at WHSmith, myfavouritemagazines.co.uk
or simply search for 'T3' in your device's App Store

SUBSCRIBE TODAY AND SAVE!
www.myfavouritemagazines.co.uk/T3

GET IT ON Google play zinio Available on the App Store

GET YOUR **FREE** GIFT!

SUBSCRIBE

& RECEIVE A FREE ECLIPSE CHARGER *

CHOOSE YOUR PACKAGE

PRINT EDITION

ONLY £32

6-MONTH PRINT SUBSCRIPTION

- Never miss an issue • Delivered to your home
- Get huge savings on the cover price

PRINT + DIGITAL

ONLY £47

6-MONTH PRINT + DIGITAL BUNDLE SUBSCRIPTION

- Never miss an issue • Delivered to your home
- Instant access on your digital device

Native Union Eclipse Charger

3-port, high-speed USB charger with cable management

GET A GIFT WORTH **£69.99**

Charge three devices

Smart IC technology ensures each device is charged at max speed with the right voltage

Keep cables neat

The integrated cable management system holds up to six metres of cable without overlapping

Looks stylish

The Eclipse rises with a double-tap, and emits a subtle glow in the dark so you can find your cables

ORDER ONLINE & SAVE

www.myfavouritemagazines.co.uk/lud/eclipse

OR CALL 0344 848 2852

QUOTE CODE **ECLIPSE** WHEN CALLING

* This offer is only available for new UK subscribers. Gift is subject to availability. Please allow up to 60 days for the delivery of your gift. In the event of stocks being exhausted we reserve the right to replace with items of similar value. Prices and savings quoted are compared to buying full-priced print issues. You will receive 13 issues in a year. You can write to us or call us to cancel your subscription within 14 days of purchase. Payment is non-refundable after the 14-day cancellation period unless exceptional circumstances apply. UK calls will cost the same as other standard fixed line numbers (starting 01 or 02) or are included as part of any inclusive or free minutes allowances (if offered by your phone tariff). For full terms and conditions please visit **www.bit.ly/magterms. Offer ends 30/09/18**

SPECIAL REPORT:

ML in security

KEY INFO
Infosecurity Europe is the largest event for information security professionals on the continent, with over 400 exhibitors and over 19,500 visitors.

While GDPR raises concerns at InfoSec Europe in London and the shift to cloud technology continues, AI, automation and machine learning gather pace

Laurent Gil started to get driving tickets issued in Los Angeles. He didn't live in LA, but the person using his driving licence number did and had obtained it from the high-profile Equifax hack. Understandably, Gil's not a fan of non-AI-based security platforms, because he says it only took one person in a 200-strong team at Equifax to forget to patch Apache Struts for his licence to be stolen. Gil also happens to work as a Product Strategy Architect at Oracle, and was at InfoSec Europe 2018 in London speaking on Machine Learning and how Oracle + Dyn uses ML for detection as part of its DDoS protection and malicious bot mitigation.

Although there was a strong preoccupation with GDPR and how to secure data at InfoSec Europe this year, AI, automation and machine learning are evidently growing trends in the information security business. This is not surprising given the industry's traditional view that the adversary tends to have all the fun. A bad actor can employ all manner of tricks and pick away at the defence, while the defenders try to make the best of limited resources and cover all attack surfaces.

Machine Learning, in particular, promises to even the playing field and companies such as open source stalwart Elastic were promoting the need to think differently and store more data for threat-hunting. "Everyone limits their data acquisitions," Neil Desai, Solutions Architect for Elastic, told us after his talk on detecting with ML. Security has lived in this peculiar silo for a long time, where limited slices of data have been used to hunt down attackers. Storing terabytes of logs just hadn't been practical for many companies, and the more data that was stored the worse the search performance became, taking hours to complete. However, times have changed and admins "need to be able to search all the data in seconds or minutes, not hours", says Desai.

Another issue for security professionals is visibility, he explains. Half the traffic on the internet is now protected by HTTPS, according to the EFF's last report in early 2017. If this didn't complicate things enough, Desai says – quoting Matt Graeber, world-

renowned pen tester – "A sufficiently advanced threat actor is indistinguishable from a competent system administrator." Which effectively means you'll be lucky if an attacker trips your security-related devices and generates security-related events. Unless you are able, or legally allowed to, decrypt SSL traffic, your chances of finding something is quite restricted, says Desai.

> "Attackers are hitting a single host and hopping laterally until they get to the information they want"

Additionally, the traditional approach of pulling logs from the data centre and the perimeter of your architecture via event, firewall, IDS and netflow logs just isn't effective any more. Desai says that attackers are hitting a single host and hopping laterally until they get to the host that either has the information they want, or access to the information they want to steal – while the admin is none the wiser because they are too far in front. The only appropriate response to such threats is to collect logs from a company's clients themselves for analysis, which could mean hundreds or thousands of workstations. In the past, searching this amount of data would be impractical and also create volume and scale issues. However, ML solutions thrive on data, and data that's stored for longer, as it enables an investigator to follow the trail, find the initial host and infection point.

Machine learning for all

Like many ML products in different fields, Elastic wants to bring ML to the masses without the need for a data scientist on staff who understands the mathematics and algorithms involved. In fact, ML aims to democratise threat-hunting. There's a recognised four-step process followed by many threat hunters, which was originally defined by Sqrrl – a company spun out of the NSA specialising in advanced threat-hunting. This involves creating: a hypothesis for an investigation; consideration of the right tools to use; what the patterns and TTPs (Tactics, Techniques and Procedures) of a cyber attacker that you're looking for might be; and some kind of automated analysis. Desai says that Elastics's ML can handle everything beyond the hypothesis, which means that threat-hunting can done by anyone once the initial fields that ML analyses are defined. Previously, says Desai, only the senior staff got to do the whole process, because they have the experience and an understanding of what's going on.

ML also goes beyond the regular search methods that an admin would do to determine if their hypothesis is correct, such as bar charts and pie charts, basic stack analysis and basic graphing. Desai says Elastics ML can uses normalisation, baseline analysis (to identify starting points) and statistical analysis on time-series events. So once the initial fields are defined, ML will come up with a baseline for determining what's normal or expected, but you don't have to know everything about the process and the algorithms involved. The use of automation, orchestration and AI are seen as significant drivers for change in security, and as companies seek to employ the right staff with the appropriate skills, they are increasingly experiencing a skills shortage. According to 2018's report from ISACA – a non-profit members association for information governance, control, security and audit professionals – 59 per cent reported having unfilled security positions, while only 25 per cent of applicants were 'well qualified' for the role. ML could effectively fill some of the gaps, but as Desai commented in regard to SOCs (Security Operations Centres), an initial alert could be chalked up as a false positive through lack of understanding by a tier-one analyst. ■

SPOTLIGHT

Fighting botnets with ML

Once he'd stopped shaking his fist at Equifax, Laurent Gil (right) presented a case study of stopping a nasty botnet attack on "a very famous media site in the UK" using Machine Learning.

AI is good at identifying attacks through similarities, but this attack "was almost impossible to identify, as they were burying a single request within millions of requests," explained Gil. First, they got lucky spotting an early-morning attack, but they wanted to see if they were any patterns to what was inside the packets. Traditional techniques didn't work in this case, such as limiting the number of requests by IP address or geolocation, as the IPs were spread across different addresses and countries. "We couldn't even look at user agents as the guys were very smart and using browsers that looked real," explained Gil.

But what finally clinched it was ML analysis of the HTTP headers: "We found that the botnet was using a particular header in the HTTP requests in a certain order." So the team were able to inject vectors into its machine learning techniques that used 126 binary features and catch all the bad actors with 100 per cent probability: "All the 14,544 malicious requests were caught." said Gil. "Even the hacker probably didn't know there was an order to the way they did the headers." For Gil this was a perfect example of what ML in security is about: "Reducing human error while enabling and enhancing human oversight."

Git: Host a Git server and GitHub alternatives

Break away from GitHub by hosting your own Git projects on a personal remote server

John Gowers

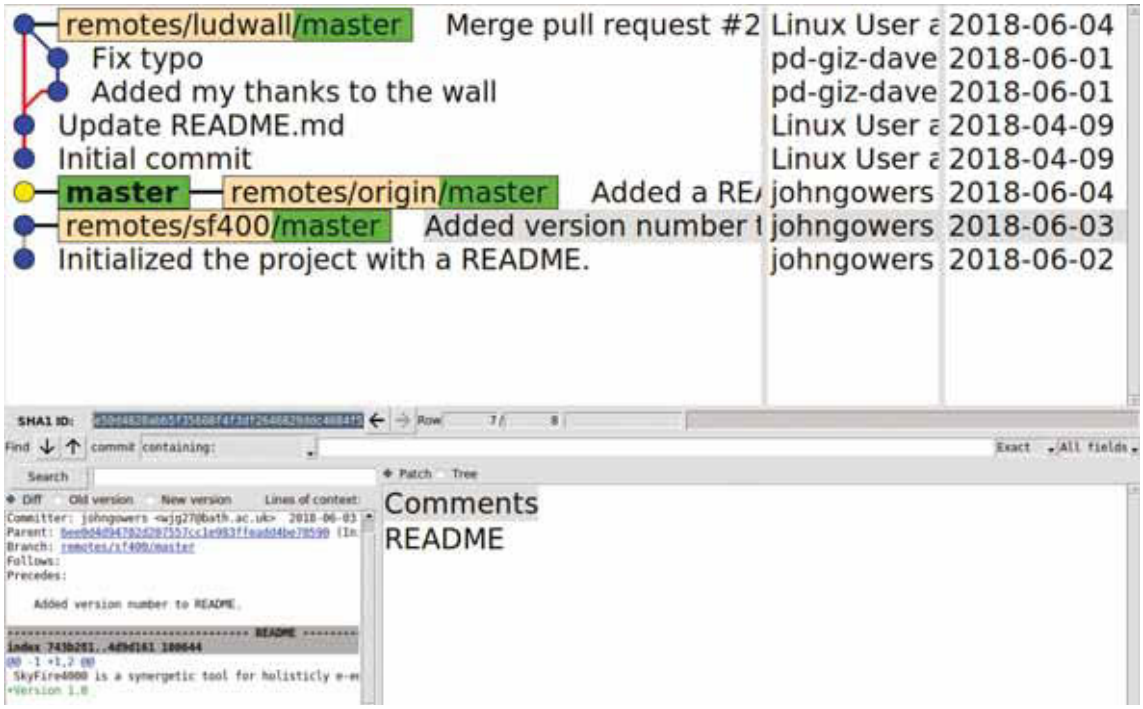
John is a university tutor in Programming and Computer Science. He likes to install Linux on every device he can get his hands on and has extensive programming experience.

Resources

■ **Git**
If not already installed, install through your package manager or from **https://git-scm.com**

■ **OpenSSH**

■ **Access to a sudo-privileged user on a remote server** (recommended but not essential)



Above We can track any number of remote repositories from a Git project. Typically, one is named origin, and is the primary place for us to push code, but more complicated setups are possible – and useful

So far, we've been using the GitHub.com site to host our Git repositories. GitHub is a great site, and it's a very quick and easy way to get a project up and running with Git. It's no surprise that it's used to host a whole range of repositories, from simple personal projects to the code for Git itself and the Linux kernel.

Nevertheless, Git is not tied to GitHub, and there are a number of reasons you might decide you want to host a Git repository on your own server. One reason is if you want to make a repository private. By default, repositories hosted on GitHub are public, so anyone can see the code. You can make a repository private, but only if you pay $7 per month for a Developer account, or $9/month for a Team account.

You might say that this is money well spent – and rightly so – but if you have space on your own server, and don't need any GitHub-specific tools, it is also money that you don't need to fork out. Self-hosting means you are in complete control over who has access to your data. It's also surprisingly easy to do, as we shall see.

Now that Microsoft has acquired GitHub, we know that many readers have objected to using the hosting platform anymore and are considering other options. Although Micosoft has changed its positon on open source software significantly in recent years, we've decided to included some alternatives to GitHub.

Simulate a remote server

This tutorial will work best if you have a remote server set up with SSH access, so that you can shell into the server by running the following command (replacing `myusername` and `myremote.example.com` with the appropriate username and hostname for your server).

```
$ ssh myusername@myremote.example.com
```

However, if you haven't got access to a remote server, you can simulate one locally. That way, at least you'll know what to do if you do ever need to set up a Git repository on a remote host. To start with, edit the `/etc/hosts` file:

```
$ sudoedit /etc/hosts
```

The file should contain something like the following:

```
127.0.0.1       localhost.localdomain
computer
::1             localhost.localdomain
computer
```

Below these lines, add the following:

```
127.0.0.1 myremote.example.com
```

This way, the computer will resolve the domain `myremote.example.com` to the local IP address `127.0.0.1`. You can test this by running `ping myremote.example.com` and checking that it returns the correct IP address. If editing `/etc/hosts` doesn't work for any reason, you should be able to substitute `localhost` for `myremote.example.com` and get the same results.

We should also set up a key pair for authorisation so that we don't need to type in our password every time we access the server. We use the `ssh-keygen` program to generate a key pair, and then add the public key to our list of known keys, following the default options as shown in **Figure 1**. If you already have a key pair set up, you can use that instead.

You should now be able to SSH into your fake remote machine as follows:

> “Paying for GitHub membership is money well spent, but if you have access to your own remote server, then it's money you don't need to fork out”

```
$ ssh -i id_rsa myusername@myremote.example.com
```

If this doesn't work, you might need to start the SSH service first. Try `sudo service ssh start`, or `sudo systemctl start sshd` if you are using Systemd.

The first time we connect, SSH will prompt us to accept the RSA fingerprint for the new host. Type `yes`.

Create a user for Git

It's a good idea to create a new user to control the Git repositories, so that later on we can allow people to push code without giving them full SSH access to the machine. Log on to your remote machine and create a new user there in the usual way, giving it a strong password. If you have followed the previous section to create a fake remote host on your local machine, you only need to create a new user on your own machine:

Figure 1

```
$ ssh-keygen
Enter file in which to save the key (/
home/user/.ssh/id_rsa):
Enter passphrase (empty for no
passphrase):
Enter same passphrase again:
$ cd ~/.ssh
$ cat id_rsa.pub >> authorized_keys
```

Left Adding public keys to ~/.ssh/authorized_keys on the remote server enables users to authenticate with their private keys rather than have to remember a password

```
$ sudo useradd --create-home gituser
$ sudo passwd gituser
```

We should be able to SSH into this new user using `ssh gituser@myremote.example.com`, typing in the password that we chose for that user. In order to set up key pair access for that user, we also need to copy our public key across to them:

```
$ < ~/.ssh/id_rsa.pub ssh gituser@myremote.
example.com "mkdir -p ~/.ssh && cat >> ~/.ssh/
authorized_keys"
```

If you are unable to create a new user for some reason (for example, if you haven't got root access to the remote machine), you can use your existing user account. Beware, though, that you won't be able to give anyone else access to the repository without giving them access to your account, so only do this if you are sure that you're the only person who is going to be using the repository.

Set up the Git server

To start with, we need to SSH on to the server where we are going to set up the Git repository, using `ssh gituser@myremote.example.com` (replacing `myremote.example.com` with your host name). Once inside, create a new directory to hold the repository, then navigate inside it like so:

```
$ mkdir -p ~/git-repos/sky-fire-4000.git
$ cd $_
```

Once inside, we run the following command to initialise a bare repository.

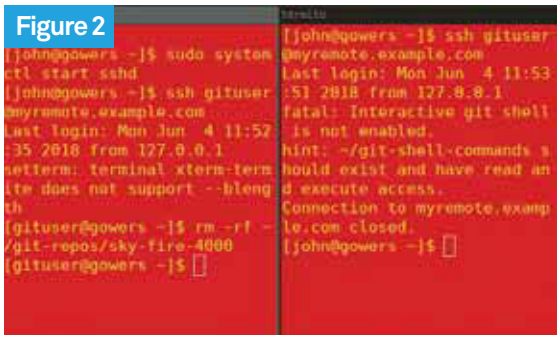
```
$ git init --bare
```

If you type `ls`, you'll see some strange files and directories inside: `branches`, `config`, `description`, `HEAD` and so on. Actually, these are not so strange: if you go inside any Git repository and look inside the hidden folder `.git`, you'll see exactly the same files and folders inside there. These folders hold all the information about the Git repository, but in a special compact form. Normally, these folders (the Index and HEAD) are hidden inside the `.git` directory, so we only see the files in the working directory when working with a GitHub repository. The `git`

Right If we give full shell access to the user account we've created, a malicious collaborator could perform destructive acts such as deleting the repository (left screen). We can fix this by changing the default shell to `git-shell` (right screen)

Far Right Calling `git init` creates the index and head, but does not affect the contents of the working directory

Figure 2



`add` and `git commit` commands take the contents of the working directory and packages them up into the special compact form.

`git push` then transfers the contents of `.git` to the remote host. Since we never expect to make changes directly on the remote host, there is no need for it to hide its main contents inside `.git`, so they are out in the open: this is what makes it 'bare'. Note that we could also have run the command `git init --bare ~/git-repos/sky-fire-4000.git`, which would have created the directory `sky-fire-4000.git` and initialised the bare repository in it in one step.

Clone the repository and make changes

That's all there is to it! Log out of the remote machine and navigate somewhere locally where we can clone the repository. We'll use `git clone` as usual; the only difference is that instead of using a GitHub address, we'll use the qualified network path to the location where we created the bare repository:

```
$ git clone gituser@myremote.example.com:/
home/gituser/git-repos/sky-fire-4000.git
Cloning into 'sky-fire-4000'...
warning: You appear to have cloned an empty
repository.
$ cd sky-fire-4000
```

If you haven't got an SSH key pair set up for the remote host, you'll need to type in a password every time you clone, push or pull from the repository. So it is definitely a good idea to have one set up: Git will automatically detect and use any SSH public keys that you've added.

Having cloned the repository and moved into it, we can add new files and make changes in the usual way – like this, for example:

```
$ cat > README
SkyFire4000 is a synergetic e-widget for
holistically-focused corporate e-businesses.
<Ctrl+D>
$ git add README
$ git commit -m "Initialized the new
repository with a README."
```

Next, we can go ahead and push the changes to the server in the usual way:

Figure 3

```
$ git init
Initialized empty Git repository in /
home/username/linux-articles/.git/
$ git status
On branch master
No commits yet
Untracked files:
  (use "git add <file>..." to include in
what will be committed)
        README
        article.txt
nothing added to commit but untracked
files present (use "git add" to track)
```

```
$ git push -u origin master
```

If we clone the repository somewhere else, the changes we've made should appear there as well.

Secure access to the repository

At the moment, we control access to the repository through the user `gituser` on the remote server. In order for collaborators to push to or pull from the repository, they need SSH access to this user, either through knowing the password or (preferably) by having their public keys added to the `.ssh/authorized_keys` file. This means that they will also be able to use SSH to log on as `gituser` and perform non-Git-related activities – see **Figure 2**.

To prevent a rogue collaborator from doing something we don't want, such as deleting the whole repository, we need to restrict the `gituser` account so that it can only perform activities that are specifically related to pushing to and pulling from our repositories. The easiest way to do this is to change the login shell for `gituser` from `bash` to the special `git-shell` shell that comes bundled with Git. `git-shell` cannot normally be used interactively, so collaborators can no longer use SSH to log in as `gituser`. All it allows are commands that are specifically related to Git repositories.

> "We often create a code base, and only later convert it into a Git project to share it. Git has a nice mechanism for doing that."

Changing the shell is easy. It's a good idea to check whether `git-shell` is added to the current list of shells by running `cat /etc/shells` on the remote server. If it

The .git extension

When creating a centralised Git repository on a remote server, the convention is to give the root directory the extension `.git`, as we've done here. When we clone the repository locally, Git automatically strips off the `.git` to give us the name for the repository. It's not compulsory to use `.git` – Git works fine without it – but it's widely used as a convention to distinguish between local repositories that are in active use and remote copies that exist only to be pushed to and pulled from.

appears, you don't need to modify `/etc/shells` further. Otherwise, you should append the absolute path to the `git-shell` executable. Since `gituser` has not got `sudo` access, you'll need to log in with a privileged account in order to carry out these next few steps:

```
$ ssh myusername@myremote.example.com
$ cat /etc/shells # If git-shell already appears, then skip the next command.
$ which git-shell | sudo tee -a /etc/shells
# Do not omit the -a, or you will lose the existing contents of /etc/shells!
```

Still logged on with our privileged user, we can now run the following command to change the shell for `gituser` to `git-shell`:

```
$ sudo chsh gituser -s $(which git-shell)
$ logout
```

The first thing that we notice is that SSH access to `gituse`r no longer works:

```
$ ssh gituser@myremote.example.com
fatal: Interactive git shell is not enabled.
hint: ~/git-shell-commands should exist and have read and execute access.
Connection to myremote.example.com closed.
```

We can still, however, perform any legitimate Git-based activities through the `gituser` user:

```
$ mkdir sandbox && cd $_
$ git clone gituser@myremote.example.com:/home/gituser/git-repos/sky-fire-4000.git
```

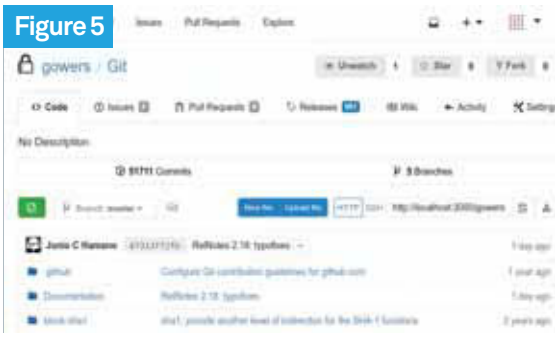
If we make some modifications, we see that we can still push them to the remote server:

```
$ echo "Version 1.0" >> README
$ git commit -a -m "Added version number to README."
$ git push -u origin master
```

This way, you can give access to `gituser` to your collaborators and they won't be able to use that user to do anything malicious or damaging beyond what is already possible through Git. It might happen that we want to access `gituser` through another shell – for example, in order to create a new repository. In that case, assuming we still have access to a `sudo`-privileged user on the remote server, we can log in with that user and then switch to `gituser` with the `su` command, using the `-s` option to change the default shell:

Figure 4

Figure 5




```
$ ssh myusername@myremote.example.com
[myusername]$ su gituser -s $(which bash)
Password:
[gituser]$ git init --bare ~/git-repos/sky-fire-8000.git
```

This means we have the best of both worlds: we have full access to the `gituser` account, but our collaborators only have the partial access provided through `git-shell`.

Convert a code base to a Git repository

So far, we have assumed that we are using Git with our repository from the beginning. In an ideal world, this would be the case, but it is probably more common that we start creating a code base, and only later decide to convert it into a Git project in order to share it with others. Git provides a nice mechanism for doing that.

Suppose we have a directory full of articles about Linux and that we want to convert it into a Git project:

```
$ mkdir linux-articles && cd $_
$ touch README article.txt
```

At this point, `linux-articles` is a directory with files in it, but it isn't a Git repository. We can fix this by running the command `git init`, as in **Figure 3**.

The command `git init` creates a `.git` file containing an empty repository, without modifying the working directory –so we could commit our changes using `git add` and `git commit`. First, though, we need to make sure that our repository is hosted somewhere, since it is currently only hosted locally. In order to do this, we need to SSH into our remote machine and create a second empty repository on there, in exactly t he same way as we illustrated above:

```
$ ssh myusername@myremote.example.com
$ su gituser -s $(which bash)
$ git init --bare ~/git-repos/linux-articles.git
```

The centre does not hold

You're not required to have one central remote repository you push to as Git is decentralised. In fact, for some projects, it might be preferable to have no centralised repository at all; in this setup, each collaborator has their own copy of the repository, and they use tools such as Git's e-mail support to transfer commits around. At the other end of the scale, it might make sense to use multiple remote repositories. We can use `git remote add/remove` to give them all descriptive names.

Top Gitea has a clean web interface, and provides useful functionality such as forks and pull requests

Left Gitea can use many different database frameworks to store its data

We now log back out of the remote host and into our local copy of the repository. At the moment, we have one empty bare repository sitting on the remote server and a second normal empty repository on the local machine. While there is code in the local working directory (namely the Linux articles that we started with), it hasn't been added to the index, so as far as Git is concerned, these are both empty repositories.

We can now run the `git remote add origin` command inside the local repository to tell Git that the remote repository is the host for the local one. From inside the `linux-articles` directory, use:

```
$ git remote add origin gituser@myremote.example.com:/home/gituser/git-repos/linux-articles
```

Technically, this command isn't doing anything special: all it does is let us use the short identifier `origin` rather than the long address. If we were to omit this command, we'd have to type the whole address every time we wanted to push something to the server:

```
$ git push -u gituser@myremote.example.com:/home/gituser/git-repos/linux-articles master
```

The `git remote` command enables us to maintain a list of remote branches with short identifiers. We can use `git remote add` to add a new address and `git remote remove` to remove an existing one. There's nothing special about the name `origin`, but it tends to be used for the server where our repository is hosted. In addition, whenever we use `git clone` to clone a repository, Git always marks the location we cloned from using the name `origin`.

Now, all that remains to do is to perform an initial commit that contains all the files that we already had in the directory, and push that to the remote server.

```
$ git add .
$ git commit -m "Initial commit of existing Linux articles."
$ git push -u origin master
```

If we want to be able to run `git pull`, we also need to tell the repository that `origin` is the default upstream location. We run the following commands for that.

```
$ git config branch.master.remote origin
$ git config branch.master.merge refs/heads/master
```

Use Gitea

Gitea (pronounced 'git tee') is an open source Git hosting package, written in Go, that makes it easier to host your own Git projects. If you miss GitHub's functionality, such as the nice graphical interface or the ability to make pull requests, you might like Gitea. Using Gitea is, in a sense, like hosting your own private version of GitHub.com: you can allow people to create user accounts, fork repositories, submit pull requests and so on. It is intended to be a lightweight solution, and can be run on simple hardware such as the Raspberry Pi.

To get started, log on to the machine where you want to host your repository. We found that the easiest way to install Gitea was to download the binary from their website like this:

Figure 6

Figure 7

Below right Finding out your public IP address requires you to connect to an external service. An alternative to using Google is to run the command `curl ipinfo.io/ip`

Bottom right GNU Savannah hosts a number of important GNU projects, as well as projects contributed by the rest of the Free Software Community

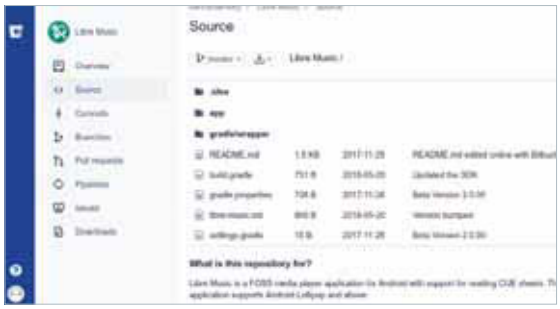
GNU Savannah

Savannah (**https://savannah.gnu.org**) is a version control site developed by the GNU foundation. As you can see from **Figure 7** the site interface is a bit different from GitHub and Gitea. To get started, you'll need to go to **https://savannah.nongnu.org/account/register.php**. Creating your own project and hosting it on Savannah is a bit harder.

Being so closely linked to the Free Software community, Savannah is committed to hosting only software projects that are free, depend only on free software and run on free operating systems. Unlike GitHub, the software that Savannah runs on is itself free software, as are all its components. This means that if you want to create a project on Savannah, you'll need to satisfy the Savannah team that your project is free software from top to toe.

There's a helpful guide to getting your Savannah project approved at **http://http://savannah.gnu.org/maintenance/HowToGetYourProjectApprovedQuickly**. It's certainly more work than starting a project on GitHub, but the process means that users can have increased confidence in your software and be sure that it is truly free. Savannah prohibits history-rewriting commits to ensure that the full version history of software is always available.

If you are comfortable with the slightly zealous focus on the Free Software movement, and if you can get your project approved, you'll be part of a devoted and prestigious software community that includes the repositories for all the GNU software that we use every day in Linux. These include things such as the Bash shell (**https://savannah.gnu.org/projects/bash**) and other essential programs such as the GNU C compiler (**https://savannah.gnu.org/projects/gcc**) – so this is big-league stuff.



```
$ wget -O gitea https://dl.gitea.io/gitea/1.4.2/gitea-1.4.2-linux-amd64
$ chmod +x gitea
```

You can then run the following to start the program:

```
$ ./gitea web
```

In a browser, navigate to **http://localhost:3000**. You should see a form as in **Figure 4**. The first thing we need to do is to set up a database for Gitea to use internally. Gitea supports MariaDB/MySQL, SQLite, TiDB and others, and it's up to you to set this up. As an example, if you have MariaDB set up and configured with a root user, you can run the following to create a database called `gitea` and a user, also called `gitea`, to administer it.

```
$ mysql -u root -p
mysql> CREATE DATABASE 'gitea' DEFAULT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_general_ci';
mysql> CREATE USER 'gitea'@'localhost' IDENTIFIED BY 'password';
mysql> GRANT ALL PRIVILEGES ON 'gitea'.* TO 'gitea'@'localhost';
mysql> \q
```

You should replace `password` with an actual password. Now go back to the page **http://localhost:3000/install** in your browser and type in the details for the user and database you've just created. You should be able to leave the other details as they are; scroll down and press 'Install' at the bottom.

This takes you to a page where you can create a user account. Create one, and you'll be taken to the main page, as shown in **Figure 5**. Here, we can create new repositories, or migrate them from GitHub or other sites.

Sharing this site with other people is a routine port-forwarding job. First, we need to find out our internal IP address. We can find that by running the `ifconfig` command, and looking for the entry corresponding to our current network connection. In our case, the first couple of lines of that entry were:

```
wlp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.7  netmask 255.255.255.0  broadcast 192.168.1.255
```

and the IP address we want is the one immediately following `inet` – 192.168.1.7. To carry out the actual port forwarding, we recommend using MiniUPNP (**http://miniupnp.free.fr**). Having installed it, run the following command to broadcast your Gitea instance on port 617, replacing 192.168.1.7 with your own internal IP address.

```
$ upnpc -a 192.168.1.7 3000 617 TCP
```

Lastly, we need to find our public IP address, which we can do by searching "what's my IP address" on Google, as in **Figure 6**. Supposing it's 2.4.6.8, for instance, if somebody now logs on to a computer running on a different network and navigates to **http://2.4.6.8:617**, they'll get to our Gitea instance, and can create their own account. Gitea does not have the same features as Git for browsing through repositories, but if we publicise the URL of our repositories to other people, they can fork them and submit pull requests exactly as on Github.

GitLab and Bitbucket

GitLab (**www.gitlab.com**) is a direct competitor to GitHub and a similar site in many ways. Unlike GitHub, the Community Edition is free and open source, and also enables you to create unlimited private repositories. GitLab has a web interface and its own hosting, like GitHub, but it is also possible to use it to self-host repositories, very similarly to what we did with Gitea in this article. If you're interested in self-hosting with GitLab, there are very good instructions available at **https://docs.gitlab.com/ce/install/README.html**. We recommend the Omnibus installation if you're using a compatible Linux distribution.

We were impressed with the range of features available in GitLab, even in the free Community Edition, and we think that most developers will find that it gives them everything they need. It's worth noting that Gitlab.com runs the Enterprise Edition, so you can get the extra features for free if you don't care about self-hosting.

Bitbucket (**https://bitbucket.org**) is another GitHub alternative, owned by Atlassian. You can use Bitbucket online for free, although you'll have to pay if you want to host your own instance (as with GitHub). Bitbucket enables you to create an unlimited number of private repositories for free, but they're limited to five contributors each.

We enjoyed using Bitbucket, though we found it was lacking in certain features: there is no option to create new branches, and the issue tracker is so-so. Part of the reason for this is that Bitbucket is intended to be integrated with other Atlassian products.

Top left Bitbucket enables you to host private repositories for free, and has great integration with products such as Jira and Trello

Above left GitLab may not have a community the size of GitHub's, but it could be a better choice, especially if you need a private repository

Kevin Wittmer

Kevin is a senior technologist and manager in the Digital Business unit of the Bosch Group. He enjoys all things Linux.

Resources

- **Installing PowerShell Core on Linux** http://bit.ly/lud_ps
- **Install Azure PowerShell on macOS or Linux** http://bit.ly/lud_azure
- **PowerShell Core** https://github.com/PowerShell/PowerShell
- **Azure Resource Manager overview** http://bit.ly/azure_ov

Azure Powershell: Manage your resources from Linux

Open source .NET Core has paved the way for Azure PowerShell to run on Linux

PowerShell, a command-line shell and associated scripting language once exclusively for the domain of Windows Server administrators and programmers, can now run on a variety of Linux distros. The flagship of this transformation is PowerShell Core, an open source project that uses .NET Core to achieve cross-platform OS support for Linux distros such as CentOS, Debian and openSUSE, as well as macOS. PowerShell's new features and cross-platform capabilities have enabled other PowerShell 'cmdlet'-based libraries, namely Azure PowerShell, to become cross-platform capable.

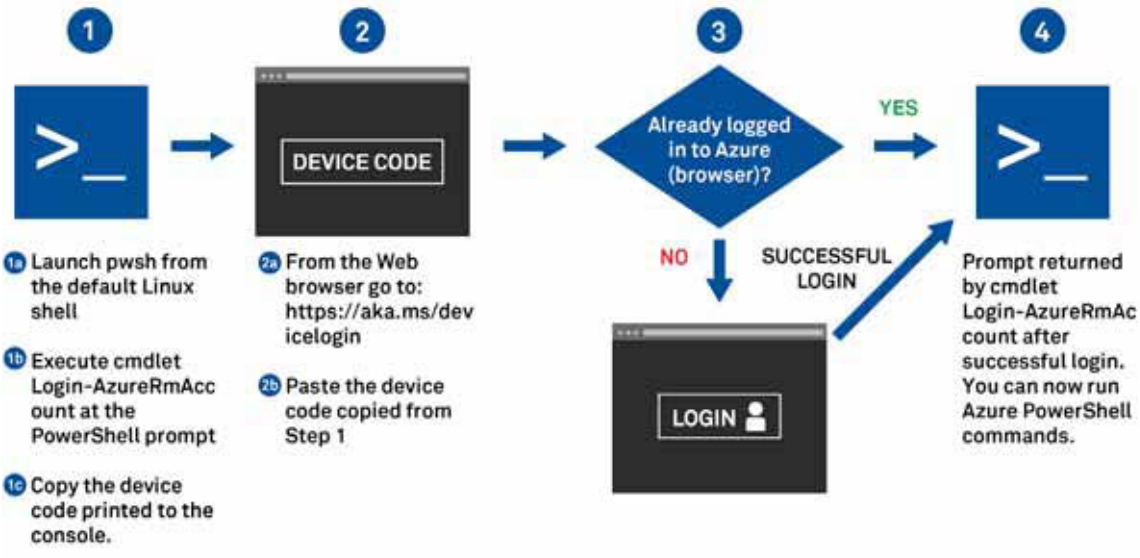
Azure PowerShell provides a broad array of cmdlets to manage a variety of Microsoft Azure cloud resources including virtual machines, app services, SQL databases, storage and networking, now with the ability to run from the Linux context. This article briefly introduces Azure PowerShell along with cmdlets to create Linux-based virtual machines in the Microsoft Azure Cloud. Along the way, we'll illustrate the virtual machine creation process with a few handy scripts – all from the Linux context!

Install PowerShell Core

To begin, check that PowerShell supports your preferred Linux distro. The box (see p41) lists supported Linux distributions where PowerShell binaries are available as installable packages using native package management commands. Note that Azure PowerShell specifies nearly the same list of supported Linux operating systems, but also includes ArchLinux and Kali Linux. For Linux distributions not officially supported, you can unpack the PowerShell binaries directly by downloading and decompressing the Linux TAR.GZ archive, or instead try out the PowerShell AppImage. A few people in the Linux community have been as daring as to use the 'Generic Linux ARM and x64' download to run PowerShell on Raspberry Pi devices, though we won't be covering that approach here.

For our example, we've chosen Debian as the target distro to install PowerShell Core. We will use Advanced Package Tool, or APT, to complete the installation. Prior to issuing the package install command, we first need to append Microsoft's package repository to the local package source configuration. This preliminary configuration sequence consists of the following steps:

1. Update the package list and install system components (`apt-transports-https`)
2. Run `curl` to import the public repository GPG keys
3. Register the correct Microsoft repository that is Linux-distro specific
4. Update (again) the package repository list to append Microsoft repo contents


1
2
3
4
DEVICE CODE
Already logged in to Azure (browser)?
YES
NO
SUCCESSFUL LOGIN
LOGIN
1a Launch pwsh from the default Linux shell
1b Execute cmdlet Login-AzureRmAccount at the PowerShell prompt
1c Copy the device code printed to the console.
2a From the Web browser go to: https://aka.ms/devicelogin
2b Paste the device code copied from Step 1
Prompt returned by cmdlet Login-AzureRmAccount after successful login. You can now run Azure PowerShell commands.

Above The command-line-based Azure PowerShell login process

Translating the above to the appropriate package manager commands, `apt` in this example, we can execute the following from our local shell environment.

```
sudo apt-get update
sudo apt-get install curl apt-transport-https
curl https://packages.microsoft.com/keys/microsoft.asc | sudo apt-key add -
sudo sh -c 'echo "deb [arch=amd64] https://packages.microsoft.com/repos/microsoft-debian-jessie-prod jessie main" > /etc/apt/sources.list.d/microsoft.list'
sudo apt-get update
```

With package sourcing now in place, the next step is to install the PowerShell binaries using your package manager's `install` command. In the case of Debian or Ubuntu, this is the well-known `apt-get install` command sequence supplied with the `-y` switch to auto-confirm prompting:

```
sudo apt-get install -y powershell
```

Once installation completes, fire up PowerShell using the `pwsh` command to establish a PowerShell session. To verify the PowerShell installation, type the following:

```
$PSVersionTable
```

This will output PSVersion information along with PSEdition, which shows as `Core`. Traditional UNIX tools are simultaneously available from PowerShell, as demonstrated by executing `cat` to print Debian OS details to console:

```
cat /etc/os-release
```

As a side note, the PowerShell Core shell program `pwsh` also has the capability to function in interview mode and serve as the default shell for your Linux instance.

Install Azure PowerShell

The next major step is to install the Azure PowerShell Core module. The PowerShell command `Install-Module` will do the work of installing the target module and associated dependencies. To install Azure PowerShell, execute the following command from your PowerShell Core session:

```
Install-Module AzureRM.NetCore
```

This results in a prompt to install PowerShell modules from an untrusted repository; answer `y` and press Return. Install-Module will then proceed to download and install various PowerShell modules that constitute Azure PowerShell. Note that 'RM' designates Resource Manager, the critical software layer in the Microsoft Azure technology stack used for deployment, management and access control of Azure cloud resources. **Figure 1** briefly introduces the five key

Figure 1

Module	Cmdlet categories of Interest
AzureRM.Profile.Netcore	Account, Context, Environment, Error, Subscription and Tenant
AzureRM.Compute.Netcore	Availability Set, Container Service, Disk, Image, Snapshot, Virtual Machine among other entities
AzureRM.Network.Netcore	Application Gateway, Route, Load Balancer, Network Interface, Security Group, Network Watcher, Virtual Network and VPN Client
AzureRM.Storage.Netcore	Storage Account and Storage Account Key
AzureRM.Resources.Netcore	AD Application, Managed Application, Resource Group and Resource Group Template (ARM)
AzureRM.Websites.Netcore	Web App

Above Azure PowerShell's five main cmdlet modules

PowerShell modules that make up Azure PowerShell. Use the cmdlet `Get-Command` to inspect the various cmdlets included in each of the modules.

For example, to inspect the cmdlets in module AzureRM.Compute.Netcore, issue the following command statement from the PowerShell command prompt:

```
Get-Command -Module AzureRM.Compute.Netcore
```

This will command output several cmdlets including `New-AzureRmVM`, which is used to create virtual machines.

Login to Azure Resource Manager

From the interactive PowerShell prompt, invoke `Login-AzureRMAccount` to login and establish an Azure PowerShell session associated with your Azure account. If you don't already have an Azure subscription, create a free account before proceeding.

```
Login-AzureRMAccount
```

This interactive method of login requires a browser session to validate the device code textually displayed during the time-sensitive external login process. The diagram on the opposite page shows the steps of the login process associated with `Login-AzureRMAccount`. Note that Azure PowerShell supports other login methods and techniques, extending to enterprise integration with Azure Active Directory (AD).

Create a Linux virtual machine

With an established Azure PowerShell session, you can now issue commands against your Azure cloud environment, such as virtual machine creation. `New-AzureRmVM` rolls-up the process of creating a new Azure virtual machine, sourcing the operating system template as an image, VM configuration or disk file. This cmdlet

Supported Linux distros

- Ubuntu
- Debian
- CentOS
- Red Hat Enterprise Linux
- Fedora
- OpenSUSE
- Generic Linux ARM and x64

Right Azure infrastructure resources use in the creation of a virtual machine

has an extensive set of command line options making it flexible enough to create required dependent infrastructure resources as necessary (see **Figure 2**), or to reference existing resources during the virtual-machine creation process.

Real-world scenarios often require that dependent infrastructure resources be instantiated and managed independently to maximise control over specific creation parameters and shared infrastructure configurations. We've created a custom script called `AzMyVmCreate.ps1`, available on the coverdisc or from FileSilo, which provides an example of creating an Ubuntu Server virtual machine with full enumeration of virtual network, IP address, subnet configuration, in-bound rules, network card, network security group and login credentials.

The cmdlets `New-AzureRmVirtualNetwork`, `New-AzureRmPublicIpAddress`, `New-AzureRmNetworkSecurityRuleConfig`, `New-AzureRmNetworkSecurityGroup`, `New-AzureRmNetworkInterface`, `New-AzureRmVMConfig` and `New-AzureRmVM` execute sequentially as part of this full enumeration example. For reference, the relevant parts of the custom script which calls these particular cmdlets looks like this:

```
$virtualNetwork = New-AzureRmVirtualNetwork
-Name $virtualNetworkName -ResourceGroupName
$uniqueGroupName -Location $groupLocation
-AddressPrefix "10.0.0.0/16" -Subnet
 $subnetConfig -Verbose
..
$publicIp = New-AzureRmPublicIpAddress
-ResourceGroupName $uniqueGroupName -Location
$groupLocation ' -AllocationMethod Static
-IdleTimeoutInMinutes 4 -Name $publicIpAddress
..
$nsg = New-AzureRmNetworkSecurityGroup
-ResourceGroupName $uniqueGroupName -Location
$groupLocation '
      -Name $networkSecurityGroupName
-SecurityRules $nsgRuleSSH
..
$nic = New-AzureRmNetworkInterface -Name
$networkInterfaceCardName -ResourceGroupName
$uniqueGroupName -Location $groupLocation '
      -SubnetId $virtualNetwork.Subnets[0].
Id -PublicIpAddressId $publicIp.Id
-NetworkSecurityGroupId $nsg.Id
```

Linux offerings available in Azure Marketplace

A variety of Linux virtual machine images are available for downloading from the Azure Marketplace. These images are prebuilt from a number of publishers. In the case of Linux this includes OpenLogic, which offers a prebuilt version of CentOS; credativ, which maintains a Debian offering in the Marketplace; and Canonical, which offers Ubuntu Server. The table to the right includes publishers and offers for common distros.

Publisher Name	Offer Name
OpenLogic	CentOS
credativ	Debian
Redhat	RHEL
SUSE	SLES
SUSE	openSUSE-Leap
Canonical	UbuntuServer

Figure 2

Virtual Machine Name	Resource Group
Virtual Network	Subnet Configuration
Public IP Address	DNS Name
Inbound Rules	Network Security Group
Network Card	Login Credentials

Note that it's also possible to create virtual machines using other Linux images; see the Linux Offerings box (far-left) and table (below-left) to learn how to identify virtual-machine SKU details for other Linux distros.

Scripting VM creation

`AzMyVmCreate.ps1` culminates in the code block shown directly below where the virtual machine configuration is instantiated and passed to `New-AzureVmVM` to do the main work of virtual machine creation:

```
..
$vmConfig = New-AzureRmVMConfig -VMName
$vmName -VMSize Standard_D1 | '
Set-AzureRmVMOperatingSystem -Linux
-ComputerName $vmName -Credential $cred | '
Set-AzureRmVMSourceImage -PublisherName
Canonical -Offer UbuntuServer -Skus 16.04-LTS
-Version latest | '
Add-AzureRmVMNetworkInterface -Id $nic.Id
..
New-AzureRmVM -ResourceGroupName
$uniqueGroupName -Location $groupLocation -VM
$vmConfig
..
```

Execute the script from your Linux-based Azure PowerShell context. The input arguments expect the following parameters:

```
param(
      [string]$resourceGroupName,
      [string]$resourceGroupLocation,
      [string]$vmComputerName,
      [string]$vmUser, [string]$vmUserPassword
)
```

To execute this script at the PowerShell prompt, type the following command:

```
./AzMyVmCreate.ps1 "WebGrpA" "NorthCentralUS"
"WebUbPZ2" "adminuser" "s0m3Passw0r3"
```

Obviously, you'll want to change the password to a more secure and personally memorable version. Upon successful execution of this script, a XML file is written to the local working directory containing the identifiers

of dependent objects captured during resource creation. The contents of the text-based XML file includes a list of identifiers of dependent objects created, including the resource group, virtual network, IP address, network interface settings and VM name. Inspect the contents using `cat`:

```
cat vui741900344-WebUbPZ2.xml
```

Figure 3 shows the contents of the text-based XML file generated by `AzMyVmCreate.ps1`.

Declarative structure

The previous example demonstrates how to create an Azure resource using imperative Azure PowerShell commands. However, Microsoft's approach to creating resources such as virtual machines is not imperative, butrather declarative.

The foundational entity of the preferred declarative approach is the Azure Resource Manager template, a JSON file that defines one or more resources to deploy to a resource group. It also defines the dependencies between the deployed resources. This template-based approach helps make it possible to more consistently and repeatedly deploy cloud resource configurations that are highly dependent and/or complex. The skeleton of a JSON template file is:

```
{
      "$schema": "http://schema.management.
azure.com/schemas/2015-01-01/deploymentTemplate.
json#",
      "contentVersion": "",
      "parameters": {  },
      "variables": {  },
      "functions": {  },
      "resources": [  ],
      "outputs": {  }
}
```

To gain further insight into the structure and composition of a Resource Manager Template file, use the PowerShell command `Export-AzureRmResourceGroup` to generate a JSON file of the virtual machine which you previously created imperatively.

```
Export-AzureRmResourceGroup
-ResourceGroupName "WebGrpA- vui741900344"
```

Operating in the Linux context, you can use any number of tools such as `vim` or `sed` to modify the JSON-based template file (shown in **Figure 4**), and then use the cmdlets `Test-AzureRmResourceGroupDeployment` and `New-AzureRmResourceGroupDeployment` to simulate and execute these changes respectively.

Best practice for imperative commands

Imperative commands are most valuable for managing the state of resources. For example, to stop or start virtual machines, Azure PowerShell offers the cmdlets

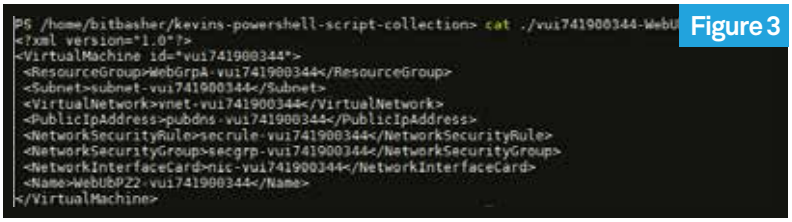


Figure 3

Figure 4

Above File output of the custom PowerShell script `AzMyVmCreate.ps1`

Left 'less' output from the cmdlet `Export-AzureRmResourceGroup`

`Start-AzureRmVM` and `Stop-AzureRmVM` for such administrative or controlling actions. To stop a running virtual machine in Azure, use a cmdlet such as `Stop-AzureRmVM` as follows:

```
Stop-AzureRmVM -Name "WebUbPZ2-vui741900344"
-ResourceGroup "WebGrpA- vui741900344" –Force
```

To transition the virtual machine from the stopped state (back) to the running state, use `Start-AzureRmVM` with a similar syntax:

```
Start-AzureRmVM -Name "WebUbPZ2-vui741900344"
-ResourceGroup "WebGrpA- vui741900344"
```

Clean up

To clean up once you've evaluated these tools – which included virtual machine creation – use the custom PowerShell script `AzMyVmDelete.ps1`, also available on the coverdisc or from FileSilo, to delete any virtual machines created, along with their base objects. This PowerShell script reads in the specified XML file and sequentially deletes resources named in the XML file.

```
AzMyVmDelete ./vui741900344-WebUbPZ2.xml
```

If your company is already using Azure as part of its cloud strategy, PowerShell Core along with the latest Azure PowerShell provides new opportunities to manage Azure resources from Linux, which many in the Linux community will find more comfortable or preferable. Keep in mind that as of this article Azure PowerShell is in beta, but stay tuned because by the time you read this it may well have shipped as a full version. In the meantime you can discuss any issues or bugs on project's GitHub site at **https://github.com/Azure/azure-powershell**.

Toni Castillo Girona

Toni has a degree in Software Engineering and a MSc in Computer Security and works as an ICT research support expert in a public university in Catalonia (Spain). Read his blog at **http://disbauxes.upc.es**.

Resources

- **Have I Been Pwned module** https://github.com/nonamed0/pam_havebeenpwned
- **VirusTotal** www.virustotal.com
- **Censys** https://censys.io/api
- **Shodan** https://developer.shodan.io
- **Koodous** https://docs.koodous.com/rest-api/getting-started
- **md5decrypt** http://md5decrypt.net/en/Api

CompSec: Use public APIs to protect your systems

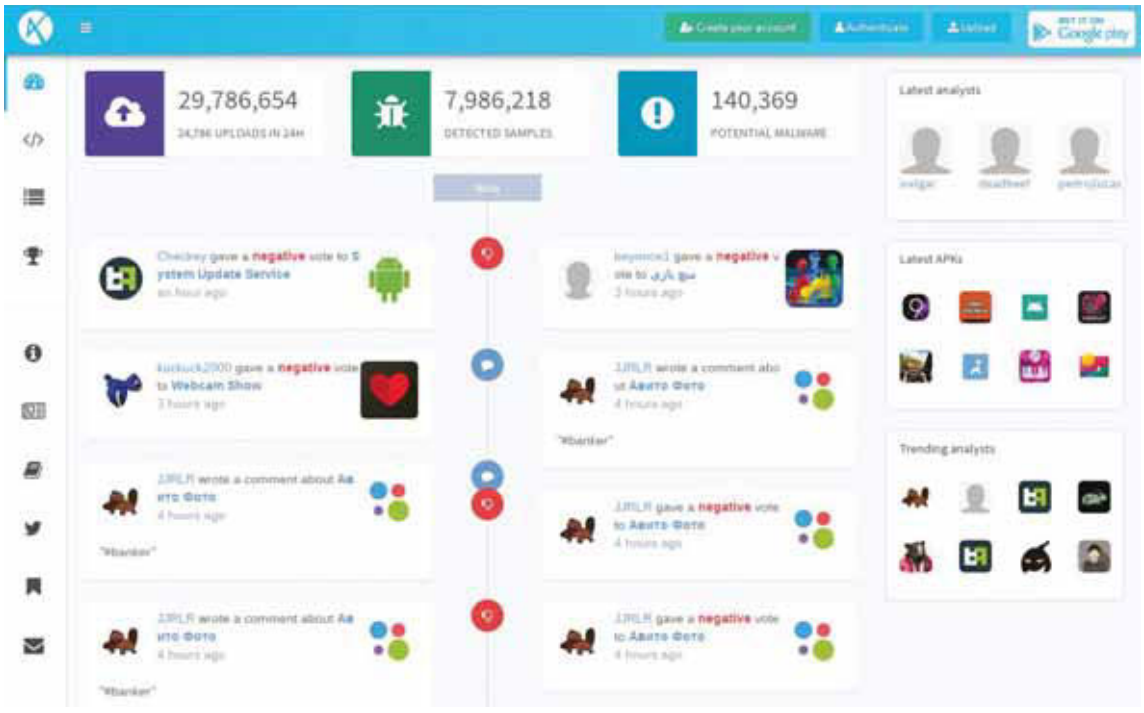
Learn how to use public REST APIs during your security engagements to feed your own scripts and tools

Data breaches, malware, malicious URLs, unknown IP addresses brute-forcing our services… Ours is a hell of a job indeed. Facing these threats alone would be suicidal, so that's why we have public APIs. These APIs are front-ends to massive databases holding vast amounts of valuable data, from cracked passwords to exposed SCADA (supervisory control and data acquisition) devices. Using these APIs is easy: more often than not all you need is a tool like curl to query them, or some basic knowledge of Bash, Python or whatever script language you fancy, in case you need more complex data-parsing.

Some of these APIs have even been integrated into some open source and commercial software, too. For example, WordPress has some plug-ins to find out whether a user password has been pwned at any point by calling the 'HaveIBeenPwned' API. Integrating some of these APIs into your own tools during your security assessments shouldn't be that difficult either. Some security tools have integrated part of these APIs already, such as Recon-ng – it integrates both Shodan and Censys APIs, among others. At some point, however, you may need to write some scripts or tools of your own, such as a PAM module to leverage the IhaveBeenPwned API, for example.

Check for pwned passwords

You've probably heard of the Have I Been Pwned? website (**https://haveibeenpwned.com**). This site is fed with gigabytes of data from different data breaches all across the internet. You can, of course, query this data using the website, or you can learn a thing or two about its public API (see Resources). Now, one interesting aspect of this API is the possibility to look for a particular password to see if it has been pwned. You may be thinking: "Hey, wait a second! I am *not* sending my password to this website just to find that out!". Us neither. That's why they devised an acceptable solution, called K-Anonymity (see **http://bit.ly/lud_kanon**). Let's imagine you want to know if the password `mariobros` has been exposed after a data breach on some vulnerable server. You compute its SHA1 hash first and then you use the first five bytes of this



Above Yes, using public APIs instead of their web interfaces is uglier – we'll give you that

hash as a prefix to the API call. All of the hashes on the website are stored in upper case, so you have to convert the lower-case hash to upper case before using the API:

```
echo -n "mariobros"|openssl sha1 \
> |tr a-z A-Z|cut -d"=" -f2
curl https://api.pwnedpasswords.com/range/C6121
```

That's it! You will get a list of possible hashes that have this prefix, along with the number of times they have been seen. Then it's up to you to locally iterate over all of these possible hashes (on average, 478) to see if there's one that matches up. Automating this is easy. We have written a simple Bash script (see **pwnedpasswordchk.sh** on the coverdisc) that can look for a particular password or for a bunch of them stored in a file (pictured, right). Get the script from the coverdisc along with the provided file `passwords.txt` and run it this way:

```
./pwnedpasswordchk.sh -f passwords.txt
```

We dare you to check some of your old passwords using this, by the way...

So far so good, but we can do better. Let's imagine you want to make sure your GNU/Linux users don't choose passwords that have been pwned. Linux Pluggable Authentication Modules (PAM) are a perfect technical solution for this. Every time a user runs the `passwd` command to change their password, a PAM module should query the API and reject this new password if it's found in the dataset. Writing PAM modules is out of the scope of this tutorial (although we will cover it soon), but suffice it to say we've implemented such a module for you. Bear in mind that messing up with PAM can

> "Public APIs are front-ends to massive databases holding vast amounts of valuable data"

easily lead to a total system lock-down or even worse, a compromised system. So don't use this module on a production system and test it on a virtual machine first.

This module is far from complete, but it will give you an idea of what you can do with public APIs. Get the module sources from the coverdisc (**havebeenpwned-pam**) and upload it to a Debian-based virtual machine. Install the necessary packages to build the module first: `apt-get install libssl-dev libcurl4-openssl-dev libpam-dev`. Our module uses curl to make the API call and OpenSSL to generate the SHA1 hash of the password typed by the user. Next, build and install the module itself: `cd havebeenpwned-pam && ./buildPam.sh`. The **buildPam.sh** script will build and copy the shared library to `/lib/x86_64-linux-gnu/security/pam_havebeenpwned.so`. On Debian-based systems, the right way to enable or disable PAM modules is by means of the pam-auth-update utility (see `man pam-auth-update`). However, our module is basically a proof of concept so in order to enable it you have to edit `/etc/pam.d/common-password` and add the following line before the first `pam_unix.so` entry by hand:

```
password requisite pam_havebeenpwned.so
```

Next, append the following option to the `pam_unix.so` entry: `try_first_password`. With these changes, our module will be the first one being executed whenever the `passwd` command runs. If it succeeds (that is, if the password has not been pwned yet), it will stack the new password with a call to `pam_set_item()`, ready to be used by the next module in line, which happens to be `pam_unix.so`. Thanks to its `try_first_password option`, this module will try first the previous stacked module's password, which happens to be the one our module has just stacked. If it satisfies it, this module will return PAM_SUCCESS and `/etc/shadow` will be updated (see `man pam_unix`). All set! Now it's testing time. Spawn a new terminal within your VM and run the `passwd` command. Type your current password and then type `mariobros` as the new one and press Return:

```
New password:
THIS PASSWORD HAS BEEN PWNED!
passwd: Authentication token manipulation error
```

Our module returns PAM_AUTHTOK_ERR and `/etc/shadow` is not updated. Try now with another password (hopefully one that has not been pwned) and this time `/etc/shadow` will be updated. In case there is an error communicating with the API – no internet connection, HTTP response error codes and so on – our module returns PAM_AUTHTOK_ERR and ends its execution without stacking this new password. Feel free to modify

Above The look on your face once you found out your password was on this database...

The REST security cheat sheet

So far you have been consuming REST APIs. At some point, though, you may be willing to develop and deploy your own. As with everything else in this field, you must adhere to good practices and ensure your API is secure by providing access control checks, encryption, input validation and so on. The OWASP REST Security Cheat Sheet is a good place to start; see **http://bit.ly/lud_rest**.

Above Ah, how beautiful JSON looks after piping it through the lightweight JQ processor

Take the challenge!
When it comes to computer security, participating in Capture the Flags (CTFs) will make you think out of the box. Although tangentially CTFs are different from real world security, we strongly encourage you to participate. We have set up a challenge for you: get it from the coverdisc (ctf/ctf.txt) and if you get stuck, read `ctf/sol/sol.txt`. Good luck!

this behaviour if you like. If you are interested in writing PAM modules, have a look at **http://bit.ly/lud_pam**.

Look for cracked hashes

As of this writing, the Md5 Decrypt website (**http://md5decrypt.net**) holds 9,950,699 cracked hashes. In order to use its API, you have to register using an email address first. If you don't fancy using one of your own emails, use one from **https://dropmail.me** instead. Go to **http://md5decrypt.net/en/Api**, enter your email address and click 'Sign Up'. You will receive your key code by email after a while. Using their API is a piece of cake. Let's imagine you want to look for the md5 hash 76657802e9335a13a3f4143907f08868; run curl like this:

```
curl "http://md5decrypt.net/en/Api/api.php?hash=\
> 76657802e9335a13a3f4143907f08868&\
> hash_type=md5&email=<YOUREMAIL>\
> &code=<YOURKEY>"
```

Of course, automating it is as easy as calling the API from curl. We have written a Bash script to use this API; get it from the coverdisc (md5decrypt.sh). You can look for one hash at a time or provide the script with a file containing a list of hashes (type:hash), one per line (see the `hashes.txt` file on the coverdisc). Give it a try:

```
./md5decrypt.sh -f hashes.txt -d 3 \
> -e <YOUREMAIL> -c <YOURCODE>
```

Identify your attack surface

Censys (**https://censys.io**) is an amazing database full of exposed devices on the internet, and it's free to query. You have to register first though, to get both your API ID and your secret keys. Go to **https://censys.io**, register

> "Messing up with PAM can easily lead to a total system lock-down or a compromised system"

and fill in the form; you will then find your keys at **https://censys.io/account/api**. Censys provides a Python library that implements its API. Install it: `pip install censys`. You can use this library to write your own Python scripts. We've written a simple script that allows you to quickly query Censys to get a list of exposed devices of your choice, along with their open ports. Get it from the coverdisc (**censysports.py**). Now, let's imagine you want to see which services and devices you have exposed on your network (say, **192.168.1.0/24**):

```
./censysports.py -u <API ID> \
> -s <SECRET> -n 192.168.1.0/24
```

You can pass any valid query to the script using the `-n` flag (see **https://censys.io/ipv4/help**), so let's do something more interesting: let's look for publicly exposed SCADA systems in Spain. Run the same command as before but modify the `-n` flag accordingly:

```
-n "tags: scada and ( location:country_code:ES \
> or location.country:ES)"
```

You can easily combine different fields to form Boolean expressions as well. Say you want to make sure there are

no more Heartbleed-vulnerable servers on your network:

```
-n "192.168.1.0/24 and tags:heartbleed"
```

Censys is great, but so is Shodan (**https://www.shodan.io**). Point your browser to **https://account.shodan.io/register**, fill in the form and you will get your API key. Now install jq, a lightweight command-line JSON processor to format the responses coming from Shodan: `apt-get install jq`. Next, use curl to call the API and pipe its output to jq to get a prettier output. Let's imagine you're interested in getting as much information about a particular host on your network as possible; run curl this way:

> To use these APIs you need a tool like curl and whatever script language you fancy

```
curl https://api.shodan.io/shodan/host/\
> <HOST_IP>?key=<YOURKEY>|jq
```

Shodan also provides different libraries and wrappers to use the API from different languages, such as Perl, Python and Go (see **https://developer.shodan.io/api/clients**), to save you the burden of using curl. If you are on a paid-API plan, you can ask Shodan to port-scan your network and even to send you an alert by email whenever new devices on your network pop up.

Fend off malware

Fighting Android malware is a bit easier thanks to Koodous (**https://koodous.com**). You can install Koodous on your Android device to proactively scan any new app you install on it, but of course they provide an API as well. Register first to get your API key; browse to **https://koodous.com/register** and then get your key from **https://koodous.com/settings/profile**. Using these APIs is mandatory nowadays due to the fact that a 'bring your own device' policy has become the norm almost everywhere. You can determine whether an APK is malicious by generating its SHA256 hash first and then calling Koodous:

```
curl -s https://api.koodous.com/apks/\
> <SHA256>/analysis|jq
```

Koodous also provides a Python library that wraps its API. This wrapper is not completely functional, so we made some changes to it. Get it from the coverdisc (pykoodous.py). Let's imagine you want to recover the analysis for a particular sample; run it using these commands:

```
./pykoodous.py -s <SHA256> -T <APIKEY> \
> get_analysis|jq
```

VirusTotal (**www.virustotal.com**) will help you identify not only malicious files, but also whether an URL may be harmful or not. Register first to get your API key. Next, use the `/url/report` API call to determine whether a particular URL may pose a threat:

```
curl "https://www.virustotal.com/vtapi/\
> v2/url/report?apikey=<APIKEY>\
> &url=<URL>&scan=1"|jq
```

You can also upload a file to VirusTotal for analysis. For files up to 32MB in size you can send the file directly through an HTTP-POST form like this:

```
curl -X POST "https://www.virustotal.com/vtapi/v2\
> /file/scan"
> --form 'apikey=<APIKEY>'\
> --form 'file=@/path/to/file'|jq
```

Of course, VirusTotal also provides different library wrappers and clients for the vast majority of languages, so you don't have to waste your time re-inventing the wheel. Try the vt Python client tool: `git clone https://github.com/doomedraven/VirusTotalApi`. Install its requisites: `cd VirusTotalApi ; pip install -r requirements.txt`. Then install the tool: `python setup.py` install. The first time you run it, it will ask you where to put the configuration file along with some basic stuff, such as what your API key is and if it is public (free) or private. Finally, you can run the tool to gather information about an URL like this: `vt –url-report <URL>`. You can also look for a particular file and see if it has already been scanned with `vt -fs <FILE>`.

WHAT NEXT?

Detect malicious traffic with MalTrail

1 Introducing MalTrail

MalTrail is a malicious-traffic detection system that uses a bunch of publicly accessible blacklists, along with some static data from AV reports and custom user-defined lists to do its job. See **http://bit.ly/lud_maltrail**.

2 Install and start the sensor

MalTrail needs Python Pcapy; install it first and then clone the repository: `apt-get install python-pcapy && git clone https://github.com/stamparm/maltrail.git`. Start the sensor: `cd maltrail; sudo python sensor.py`. It updates its data on first run.

3 Start the server

You can install MalTrail on another computer to act as the server or you can use the same computer for both services. Start the server with `sudo python server.py`. Your computer will be listening on TCP port 8338.

4 Access the interface

Open your browser and navigate to **http://127.0.0.1:8338**; make sure to change the default 'admin:changeme!' credentials. Any malicious traffic detected by the sensor will be automatically logged and shown here.

5 Keep an eye on your system

Use your computer in the usual way for a while, making sure the sensor is running. Connect to the reporting interface from time to time to see if anything is triggered.

Joey Bernard

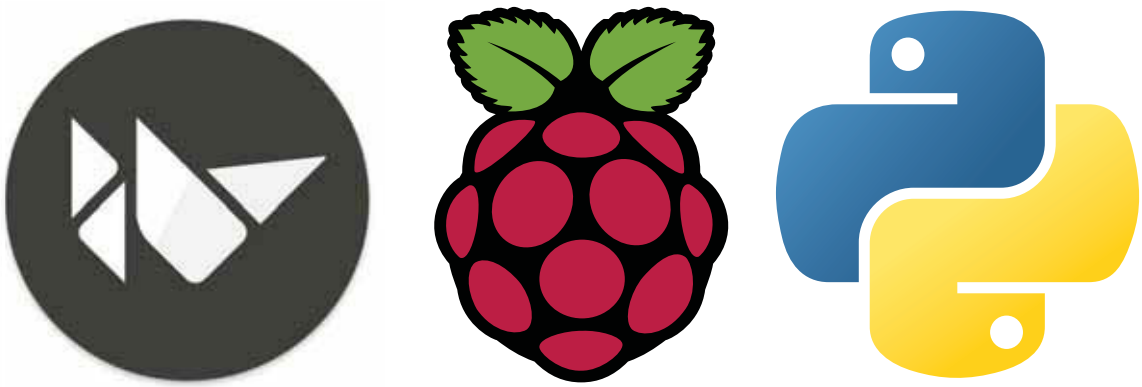
Joey is a columnist covering scientific software, as well writing Python code for the Raspberry Pi. In his day job, he helps researchers and students at the university level design and run HPC projects on supercomputing clusters.

Resources

- **Kivy** https://kivy.org
- **Pygments** http://pygments.org

Kivy: Give your code a friendly graphical interface

Many projects could benefit from a GUI dashboard – and you can create one using Kivy on your Raspberry Pi



Over the previous issues, we've looked at a lot of different ways that you could use your Raspberry Pi. But, while we've developed code that was specific for each project – such as how to talk to Bluetooth devices or communicate over the GPIO bus – we haven't really covered how to interact with people. This issue, therefore, we'll take a look at Kivy, which is ideal for creating graphical displays and dashboards to make it easier for your users to interact with your apps.

Kivy is a multi-platform library for Python that gives you the tools you need to build a GUI for your Python projects. As well as the Pi, Kivy is available on Linux, Windows, macOS and even Android – so you could take the examples given here and easily reuse them in lots of other projects.

Along with dashboard displays, Kivy provides several different interface options, such as buttons, touch-screens and even multi-touch interfaces. By the end of this article, then, you should have the tools you need to add a really useful (and attractive) interface to your Raspberry Pi project.

Install Kivy

Installation is pretty easy if you are using Raspbian, or some other Debian variant, on your Raspberry Pi. You can install it with the following command:

```
sudo apt-get install python-kivy
```

If you've moved to writing code using Python 3, install the package `python3-kivy` instead. There is also a package called `python-kivy-examples`, which provides a number of sample pieces of code, unsurprisingly. These are handy to have locally on your development machine and provide code snippets without having to be online. They are located in the directory `/usr/share/kivy-examples`. Take a look for excellent examples of all of the things you could add to your own program.

Hello world

Since Kivy is a full GUI toolkit, it will involve a bit more boilerplate code wrapping the actual functional bits. For example, the code below is an example of a classic 'Hello World' program:

```
from kivy.app import App
from kivy.uix.label import Label
class MyApp(App):
    def build(self):
        return Label(text='Hello world')
if __name__ == '__main__':
    MyApp().run()
```

As you can see, the first step is to import the `App` class and implement your own subclass based on it. In this case, the subclass is called `MyApp`. The primary method that you must implement is the `build()` function. This method gets executed when your app starts and generates the actual windows and other display elements. In the above example, the code simply generates a `Label` object in order to show the text `Hello world`. Save the code to a Python script file – `myapp.py` for example – and

run it with `python myapp.py`. Moving beyond a simple text display, what tools does Kivy provide for displaying data? Since we're developing graphical displays, you probably want to start with being able to display images in your program. This is handled through the `Image` class. For example, you could add the following to the earlier boilerplate to load an image:

```
from kivy.uix.image import Image
....
    def build(self):
        return Image(source='image.png')
```

This loads up and displays the image file given as the source for the `Image` class. This works great for images stored locally, but you may want to load images from some remote source, such as a web server. In these cases, you may want to use the `AsyncImage` class instead. This puts the loading of the image into the background and keeps the rest of your program from blocking while the image is being loaded.

Working with images

There are quite a few parameters to the class instantiation that help control the image display. You can set `allow_stretch` to tell the image fill the containing box. The image ratio is kept by default, but you could allow it to be stretched regardless by setting the parameter `keep_ratio` to `False`. If the image you're loading is an animated GIF file, Kivy gives you control over that too. You can set how many times to loop through it with the parameter `anim_loop`, and control how fast the animation runs with the parameter `anim_delay`. If your system is actually updating the image file – while you're updating a plot with new data, for example – you can tell your program to reload the new image data with the method `reload()`.

> "Since Kivy is a full GUI toolkit, it will involve a bit more boilerplate code wrapping the actual functional bits"

You may, instead, need to build up your graphical imagery from scratch. The subsection `kivy.graphics` contains a number of graphical primitives which you can use for this. These include objects such as lines, ellipses, points, rectangles, Bézier curves and triangles. There are also a number of methods available to alter these graphical elements. These include transformational functions such as translation, scaling and rotation.

In addition you have other standard graphical elements, such as pop-ups and labels, that enable you to display textual information. For example, the following bit of code displays the contents of a text file:

Above Kivy makes smooth multi-image, multi-touch displays really easy to code

```
from kivy.uix.label import Label
text_file = open('my_file.txt')
text_str = text_file.read()
label1 = Label(text=text_str)
```

You can control the layout of the text with parameters such as `font_name`, `font_size`, `color`, `halign` and `valign`. You can even make your text box interactive by adding the markup `[ref=XXXX]` and `[/ref]`, where `XXXX` is a label for the reference. This makes the text surrounded by these delineators into a link that can have some function bound to it. The following code shows how you would get the function `my_func` to fire when the word 'hello' is clicked:

```
label1 = Label(text='[ref=hello]Hello[/ref]
World', markup=True)
label1.bind(on_ref_press=my_func)
```

As you can see, you also need to include the parameter `markup=True` so the markup tags get processed correctly.

The last big graphical element that you can add to your program is the ability to play video files. All of the really hard work is wrapped up in a very complete `VideoPlayer` object. With the following code, you get a video display with all of the usual playback controls.

```
from kivy.uix.videoplayer import VideoPlayer
player = VideoPlayer(source='my_film.avi',
state='play')
```

This loads the video file and starts it playing immediately; all the standard playback controls are included.

Use the layout class

While we've looked at a lot of the elements that can be used to display things within your program, we still

Do it on your desktop

Using Kivy as your GUI toolkit, you can easily do your code development on your desktop. You can then move it off to your Raspberry Pi project, usually with no code changes at all. For people used to a descriptive GUI language like glade, Kivy provides the Kv language that behaves in the same way.

haven't examined how they're actually laid out on the screen. As with many other graphical toolkits, Kivy gives you a set of classes, called layouts, that manage the physical organisation of the various elements. As a basic example, the following code draws two buttons, one above the other.

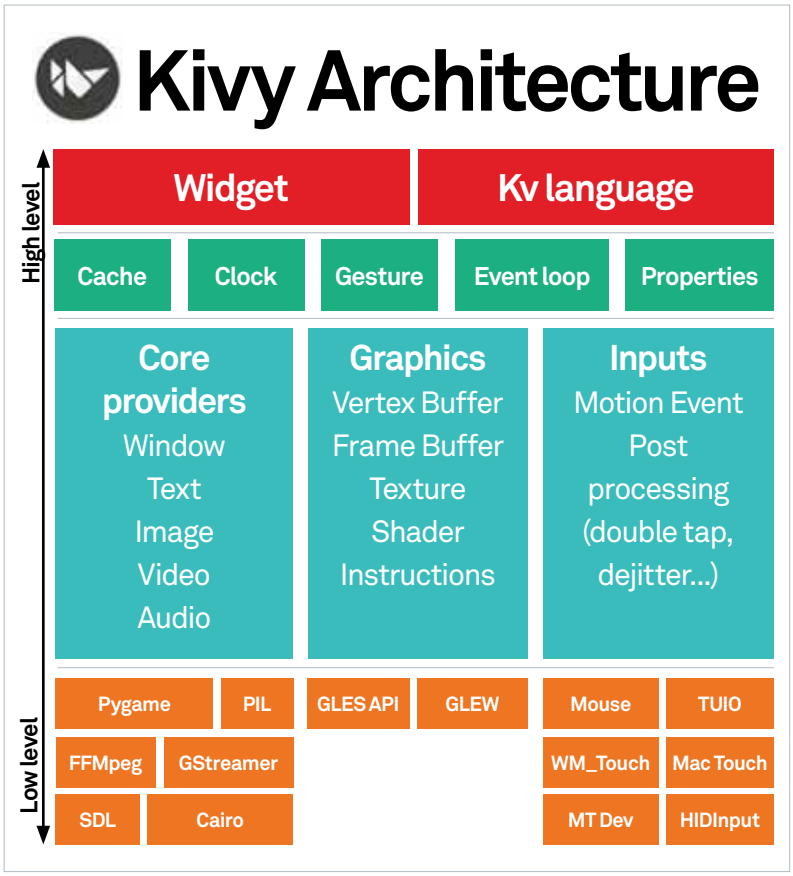
```
layout = BoxLayout(orientation='vertical')
btn1 = Button(text='Hello')
btn2 = Button(text='World')
layout.add_widget(btn1)
layout.add_widget(btn2)
```

The first line creates the layout object, orientated vertically. Then two button objects are created. All of the objects that generate graphical elements for your program subclass the basic class for widgets, so you can use the `add_widget()` method of the `layout` object to add them to the display. The `boxlayout` can be thought of as a one-dimensional layout, where you can have your widgets grouped either vertically or horizontally. Moving up one dimension, you use the grid layout object.

Unlike many other graphical toolkits, you do not have direct control over where widgets are placed within the grid; they get placed simply in the order they're added to the layout object, in row-first order. You need to give the grid layout at least one `cols` or `rows` parameter, and the sizing of the widgets need to match. That is, all of the widgets in a given column need to be the same width and any widgets in a given row need to be the same height. If you want to be more flexible in the sizing of your widgets, you may want to use the `stack` layout instead. In a stack layout, widgets in a given row or column no longer need to be the same height or width.

You can move up another dimension by using the `page` layout object. This creates a multi-page layout where widgets can be added to each page, and you can flip back and forth between them using the borders. A more structured layout object is the `anchor` layout. In this case, you define an `anchor_x` and `anchor_y` parameter, defined as top, centre or bottom. This essentially pins the children of the anchor layout to one of the borders.

> "As with many other graphical toolkits, Kivy gives you a set of classes, called layouts, that manage the physical organisation of various elements"



Below Kivy builds up from the low-level interface libraries to high-level objects such as widgets

Interacting with the world

Many of your projects will also need to accept input from users. This could come in a variety of forms – we've already seen buttons in some of the earlier code examples. Moving beyond the button, Kivy provides a toggle button, which behaves in much the same way as a tickbox. You can make it behave more like a set of radio buttons using the `group` parameter. In this case, only one member of the group can be in the down state (that is, selected) at a time.

While these types of input values are simple on or off values, that may not be enough. In these cases, you can add a slider to your program.

```
from kivy.uix.slider import Slider
s = Slider(min=-100, max=100)
```

You can read the value that the user has selected by checking the `value` property of the slider. You can also set this `value` property, which will reset the slider to that position on the display.

For the inputting of textual data, you have a couple of options. The first is the `TextInput` class, which enables you to add a text box into which the user can type. The class handles cursor movements within the text box, as well as clipboard interaction with the operating system. If you need a more robust textual input option, you may want to look at the `CodeInput` class. This lets you embed, essentially, an IDE within your code. You'll also need to have the Python module `pygments` installed, as it

provides highlighting for the widget. Any of the languages supported by pygments gets highlighted within the code editor window. For example, the following code creates a code input widget that highlights Cython code.

```
from kivy.uix.codeinput import CodeInput
from pygments.lexers import CythonLexer
codeinput = CodeInput(lexer=CythonLexer())
```

There are a huge number of 'lexers' available with pygments, but in the rare instances where you need a lexer for some obscure language, you can always build your own.

Manage events

We've looked at all sorts of display options, as well as ways that users can input their own data or interact with your program. But how do you manage the actions triggered by these interactions? Whenever you actually do anything, such as clicking a button or typing text, events are triggered within Kivy. Widgets all implement the `EventDispatcher` to manage these events in a background thread of execution. As you've seen above, events are managed by callback functions that are bound to one or more of these actions. There are several available as part of the `widget` class. For example, the following code notifies you of any changes to the state of a button.

```
def callback(instance, value):
    print('My button <%s> state is <%s>' %
(instance, value))
btn1 = Button(text='Hello world 1')
btn1.bind(state=callback)
```

This binds the callback function to the `state` property of the button, so that any time it changes, the callback function gets fired. You can even create your own with the `register_event_type()` method of the `widget` class.

Along with these class events, there are also input events; things like mouse clicks, touches and scroll-wheel events are all handled by the class `MotionEvent`. These are dispatched by the `on_motion()` method of the `Window` class. They then generate `on_touch_down()`, `on_touch_move()` and `on_touch_up()` events in the `Widget` class. You can override any of these event handlers; the code below is an example.

```
def on_touch_down(self, touch):
    if super(OurClassName, self).on_touch_
down(touch):
        return True
    if not self.collide_point(touch.x,
touch.y):
        return False
    print('You touched me!')
        return True
```

You should have noticed the references to `super` in the above example. This is because when you override one these event handlers, you become responsible for all of the required functionality. For the sections that you are not interested in overriding, you can use `super` to use the code from the base class.

The last category of events are those generated by the `Clock` object. Because of the way Kivy's event loop manages event handlers, you can't have any kind of infinite loop in you code that executes some code repeatedly, because this will end up blocking the rest of the system.

But you may have a legitimate reason for wanting to repeat a chunk of code in your event handler. If so, the correct way to do this is with the Clock object. With Clock, you can schedule function calls for some point in the future, either once or repeatedly. For example, the following code fires your callback function every second.

```
def my_callback(dt):
    pass
Clock.schedule_interval(my_callback, 1.0)
```

In this example, the callback function doesn't actually do anything. The parameter `dt` is the delta time, and is always handed in to your callback function.

Non-widget classes

There are several other classes available that provide nice-to-have functionality for your code. The `Animation` class provides a set of pre-packaged transitions that smoothly move widgets from one state to another. For example, you could have the text for a `Label` object fade into view, without having to write and debug a bunch of your own code.

We didn't get a chance to talk about textures here, but there is an included `Atlas` class that enables you to bundle a number of textures together into a single image file. This helps simplify image downloading, for example, among other things.

The last extra class is `UrlRequest`. This provides a way to interact with things on the internet outside of the main loop. This way you can do something such as downloading a file from a website without blocking your entire interface while you are waiting for it finish.

Above Kivy's examples provide a showcase application that lets you see almost all of the widgets available

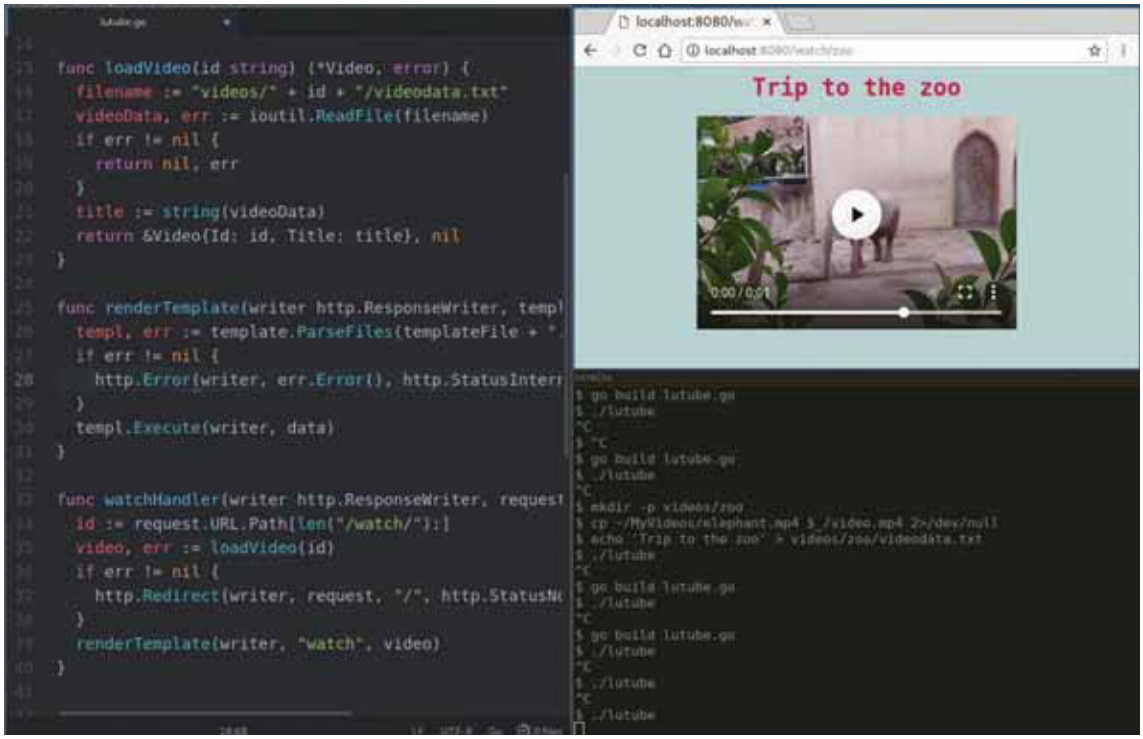
Using env variables

You can easily control most of Kivy's functionality through the use of environment variables. For example, you could set `KIVY_WINDOW` to `x11`. This will force Kivy to use the X11 windowing environment. You also have control over audio, camera usage and the OpenGL implementation to use, among other elements. These options can be placed within a configuration file, too.

PART ONE

Go: Program your own video-sharing server

Get to grips with Go, the open-source language developed and used by Google



Above Use Go to get a dynamic video-sharing server up and running

John Gowers

John is a university tutor in Programming and Computer Science. He likes to install Linux on every device he can get his hands on and has extensive programming experience.

Resources

Go
Download through your package manager or from **https://golang.org/dl**

A web browser
Ideally Chrome 4.0+, Firefox 3.5+ or Opera 25+

Until about ten years ago, nearly all of Google's server code was written in C++. While C++ programs run extremely quickly, they are also prone to a number of problems, such as the lack of built-in support for concurrency and slow build times caused in part by the way C++ handles dependencies between modules.

Google released the Go language in 2009. It was intended to resolve some of the problems it was having with C++ code, speeding up build times and leading to a more readable and testable code base. It has had a lot of success on this front: in 2012, Brad Fitzpatrick (the creator of LiveJournal and now a Google employee) decided to rewrite the dl.google.com server – from which users download Google Chrome – in Go, expecting to take a small performance hit in exchange for a more readable code base. In fact, not only was the resulting code much more readable, but when it was compiled and run, it ran just as fast as the C++ original, and actually used slightly less memory.

Go is not meant to incorporate hundreds of fashionable language features; it has been designed to be the best tool for creating large code-bases of the type generally used at Google.

About this project

Go is an open-source language, and it's very easy to get started with it. The Thanksgiving 2011 Google Doodle, which features a turkey with interchangeable feathers and which was written entirely in Go, was deployed in under 24 hours by a Google employee who was a total newcomer to the language.

One Google site that relies particularly heavily on Go is YouTube, so we thought it would be fun to create our own version: LU-Tube, a social video-sharing site for Linux Users. Over the course of the next three articles, we'll be developing a simple web server that will enable users to watch, search and upload videos, all of it written in Go. Each month, we'll release a working copy of code from

the previous issue, so that you can get stuck in even if you haven't followed the previous articles. Since this is the first issue in the series, we'll be starting from scratch.

We won't be assuming any Go knowledge, but it would help to be familiar with programming in a language such as C or C++ and to know a bit about web applications.

Base types

To get started, we'll create a directory called `lutube` and move into it. This is where our server will live. The videos themselves will live inside a directory named `videos`.

```
$ mkdir lutube && cd $_
$ mkdir videos
```

Create a new file called `lutube.go` inside the `lutube` directory and open it inside a text editor. Add the following line at the top; this will tell Go that this file contains the `main` function that should be run at the start of the program.

```
package main
```

Before we write anything, let's talk a bit about how we're going to store and represent videos. A video on our site is more than the clip itself; it's the title, and later on the keywords, the user who uploaded it and so on. We'll give each video a unique ID, corresponding to a subdirectory of the `videos` directory. Inside that subdirectory will live

> Go has no special error-catching syntax; instead, errors are passed around as ordinary values

the video file itself, plus other files storing any additional information. For now, let's create a Go `struct` that will encode a video. To begin with we'll stick to the video's ID and its title.

```
type Video struct {
  Id string
  Title string
}
```

You'll notice a few differences between this and the way that `struct`s are declared in C. Perhaps the most important is that in Go we always put the type of a variable or field *after* the variable name. In some ways, this is similar to the syntax in languages such as Haskell, Ada or Rust, where we'd write `Id: string`; in Go, we drop the colon.

We'll store the title inside a file called `videodata.txt`, living inside the video's directory. Let's write a function that will take in an ID and return a `Video` struct, with the `Title` field filled in by looking at the `videodata.txt` file.

Figure 1

```
func loadVideo(id string) (*Video, error) {
  filename := id + "/videodata.txt"
  videoData, err := ioutil.ReadFile(filename)
  if err != nil {
    return nil, err
  }
  title := string(videoData)
```

Above Error checking in Go does not involve any specialised language features beyond simple `if`-statements and function passing

```
func loadVideo(id string) *Video {
  filename := "/videos/" + id + "/videodata.
txt"
  videoData, _ := ioutil.ReadFile(filename)
  title := string(videoData)
  return &Video{Id: id, Title: title}
}
```

This is our first example of a function written in Go. You'll notice that Go uses the keyword `func` to declare a function, and that it puts the return type of the function (in this case, a pointer to a `Video` struct) at the very end of the line. We can see a few other examples of typical Go syntax; for example, we often use the symbol `:=` to create and initialise a variable on the fly. The type of the variable is inferred from its value.

The third line is quite interesting, and demonstrates that Go functions are allowed to return multiple values. In this case, the `ioutil.ReadFile` function returns both the text of the file and a value that indicates whether or not the function has encountered an error.

We've used the symbol _ for this error value, signifying that we're not going to use it. That's not great programming practice, so let's modify our function so that it behaves appropriately in case there's an error. First, we'll change the signature of the `loadVideo` function so that it too can return an error value:

```
func loadVideo(id string) (*Video, error) {
```

Then, we'll read in the error value; if there is an error, then we will return it:

```
videoData, err := ioutil.ReadFile(filename)
if err != nil {
  return nil, err
}
```

The value `nil` is used to show the absence of a value. Go has no special error-catching syntax; instead, errors are passed around as ordinary values. In this case, we are saying: if there is an error, then return `nil` for the `Video struct`, and return the error value as well.

We should also modify our existing `return` statement so that it returns a `nil` error value if everything has gone well, like this:

Right Go makes it easy to get a dynamic server up and running

```
return &Video{Id: id, Title: title), nil
```

The full function is shown in **Figure 1**. We need to do one last thing to make it run properly, which is to add a line at the top of the file, just below `package main`:

```
import "io/ioutil"
```

This is because we have used the `ReadFile` function from the `ioutil` library.

Set up the web server

Now that we've created our base type for handling videos, let's switch focus and set up a running web server. First, we'll create a `main` function that will run when the program starts:

```
func main() {
   http.HandleFunc("/watch/", watchHandler)
   log.Fatal(http.ListenAndServe(":8080", nil))
}
```

The first statement in this function tells us to identify the URL path `/watch/` with the (as yet unwritten) function `watchHandler`. The call to `http.ListenAndServe` serves on to port `8080` on our system. The effect is that when we open a browser and navigate to **http://localhost:8080/watch**, our server will call the `watchHandler` function in order to determine what HTTP response to send.

So that we can see this in action, let's quickly write an initial version of the `watchHandler` function:

```
func watchHandler(writer http.ResponseWriter, request *http.Request) {
   writer.Header().Set("Content-Type", "text/html")
   fmt.Fprintf(writer, "%s" "<h1>Under construction.</h1>")
}
```

Right It's considered better practice to group `import` statements like this, rather than write them out separately

Figure 2

```
import (
   "io/ioutil"
   "net/http"
   "fmt"
   "log"
)
```

Importing packages

In most programming languages, it is very difficult to keep track of which packages rely on which other ones. In C++, it is difficult to know when a particualr `#include` directive is necessary. We can delete the line, of course, but if the code then compiles it doesn't necessarily mean that the module wasn't used: it might be `#included` by some other header file. Go forces developers to tackle these problems early and often, so there is as little confusion as possible later on.

In Go, for example, an unused `import` is not just wasteful, but will lead to the compiler rejecting the program outright. In that way, we ensure that the packages we import are precisely the ones we need to run our program, helping to keep binaries as small as possible.

Another important feature is that Go forbids two-way dependencies between packages, preventing loops that could lead to much larger executable sizes. Lastly, the way that Go imports binaries is much more efficient than in C++: a dependency is only loaded once, even if it is pulled in by multiple packages.

Figure 3

Before we compile and run the code, the last step is to add some extra `import`s to cover some of the functionality we've used. We need to add imports for `"net/http"` to cover the networking code, and for `"log"` and `"fmt"` to cover some of the other functions we've used. We could add these as separate `import` statements, but Go allows us to combine them into one. Replace the line `import "io/ioutil"` at the top with the compound statement shown in **Figure 2**.

Now we are ready to compile and run our code! Open a command window and navigate to the `lutube` directory. Compile and run the program:

```
$ go build lutube.go
$ ./lutube
```

Now open a web browser and point it to **http://localhost:8080/watch**. The text 'Under construction.' appears on the page, as in **Figure 3**. When you get bored with this page, press **Ctrl+C** on the terminal to close the server so we can move on to the next step.

HTML templates

This is all very exciting, but we still can't watch any videos. What we want to do is programatically construct an HTML page that will contain the video itself, plus its title. We could construct this in a number of different ways, but probably the neatest way is to use Go's HTML template package.

In Go, an HTML template is a special HTML file where some sections are left blank, to be filled in later. For example, we can use the following template to display the title of our video.

```
<h1>{{.Title}}</h1>
```

Here, the `<h1>` tags are normal HTML code. The `{{.Title}}` in the middle is a placeholder for a field named `Title`. In order to render this template, we'll call the `template.ParseFiles` and `Execute` functions, passing in a pointer to a `Video struct`. These functions will read the `Title` field of that `struct` and insert it in place of the placeholder, appropriately enough.

Let's see how this works in practice. First, we'll create a directory that will simulate a video:

```
$ mkdir videos/zoo
$ echo "Trip to the zoo" > videos/zoo/videodata.txt
$ touch videos/zoo/video.mp4
```

We have used `touch` to create files that will stand in for actual video files. If you have a video on your computer in MP4 format, then copy it into the `zoo` directory and rename it to `video.mp4`. (If you're wondering why we chose that title, look up the history of YouTube.)

We can now call `loadVideo("test")` to load this into a `Video struct` that contains the ID `test` and the title `"Trip to the zoo"` that is contained in the `/videodata.txt` file.

Before we go back to programming in Go, let's create the HTML template for our video page. Create a file called `watch.html` in the `lutube` directory containing the following HTML code:

```
<h1>{{.Title}}</h1>
<video src="/videos/{{.Id}}/video.mp4" controls="controls">
   Your browser does not support the video tag.
</video>
```

In our example, once we serve the page the `{{.Title}}` placeholder will be replaced with `Trip to the zoo` and the `{{.Id}}` placeholder with `zoo`, giving the path `zoo/video.mp4`.

The slash `/` before `{{.Id}}` is important: it means that the browser fetches the video relative to the root directory, rather than relative to the current URL `/watch/zoo` (which doesn't correspond to an actual folder). All that remains is to write the code that will parse and display this code. Go back to `lutube.go` and replace the code inside the `watchHandler` function with the following four lines.

Figure 4

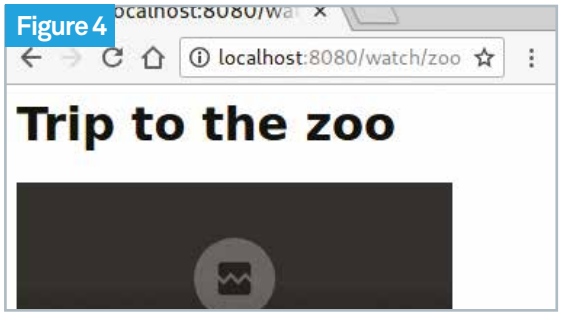


Figure 5

```
func getAvailableVideos() ([]string, error) {
    videoDirectories, err := ioutil.ReadDir("videos")
    if err != nil { return nil, err }
    availableVideos := make([]string, 0)
    for _, f := range videoDirectories {
    availableVideos = append(availableVideos, f.Name())
    }
    return availableVideos, nil
}
```

Above Go arrays are similar to the arrays in C, but they are much easier to deal with thanks to a number of functions

```
id := request.URL.Path[len("/watch/"):]
video, _ := loadVideo(id)
templ, _ := template.ParseFiles("watch.html")
templ.Execute(writer, video)
```

There are a few new things in this code. In the first line, we use a special Go feature that enables us to take an existing array and cut out a particular range of indices. The resulting object, called a 'slice', occupies the same area in storage as the original array elements, but may be used as though it were an array in its own right. For example, if `a` is an array, then `a[1:2]` cuts out the second and third elements of `a` (since indices start at `0`). In this case, we want to take the URL path (`/watch/zoo/`) and remove the `/watch/` part at the beginning, so we slice out all indices starting from the length of the string `"/watch/"` up to the end of the URL.

> What we want to do is programatically construct an HTML page that will contain the video itself, plus its title

The rest of the code loads the video object, and uses it to render the HTML template. Since we pass `video` in to the `Execute` function at the end, this is the object whose `Title` and `Id` fields will be used to populate the placeholders in the template.

We'll be ready to go as soon as we've sorted out our `import` statement at the top. We need to add `"html/template"` in order to unlock the HTML template functionality. We also need to remove `"fmt"`, because unused imports are a compiler error in Go. Once we've fixed that up, we can run `go build lutube.go` and `./`

Left We can use HTML templates to construct pages dynamically. The HTML5 `<video>` tag makes it very easy to embed videos

Naming in Go code

Go is unusual in that the names of functions and variables have a distinct syntactic meaning. Specifically, if an identifier begins with a capital letter, it's visible to other packages that import the package it lives in. If it begins with a lower case letter, it's only visible in its own package. This is exactly the same as the `public` and `private` keywords in C++, except that now we can tell at a glance whether something is public or private without going to its definition.

Format your code

One of the aims of the Go community is that all Go code should be formatted in a uniform style. To this end, Go comes bundled with a tool, `gofmt`, that automatically formats your code into the desired style. You can run it with `$ gofmt -w mycode.go`.

The benefit of having a uniform style is that everyone is free to concentrate on the code itself: no more worrying about switching between styles or arguing about brace position!

`lutube` again to compile and start the server. If you open up a web browser and navigate to **http://localhost:8080/watch/zoo**, you should see a page as in **Figure 4**.

Serve static requests

In the page shown in **Figure 4**, the video itself is blank. If you used `touch video.mp4` to create a fake video, this won't be surprising, but if you did use an actual video file, you'll be understandably disappointed that it isn't now playing, for some reason.

If you view the source of the page in your browser, you'll see that the `<video>` tag is pointing to `/videos/zoo/video.mp4`, which is the video file that we want to play. The reason it isn't playing is that we haven't yet set up the server to serve URLs starting with the `/videos/` prefix.

In this case, we want our server to act a bit more like a standard static web server that converts URLs straight into locations in a file system, serving up the required files. This is called serving a 'static request'. We could write a separate function to serve files in the `videos` directory statically, but there's an easier way. Add the following line to the `main` function, below the existing `http.HandleFunc` call:

```
http.Handle("/videos/", http.FileServer(http.Dir(".")))
```

`http.FileServer` is a special object called a 'handler' that can be used to deal with HTTP requests directly. In this case, it converts the URL into a directory name based in the root directory `.`.

After adding this new line, we can compile and run the server again. This time, when we load **http://localhost:8080/watch/zoo** in a browser, the video (if we are using one) should appear, as in the main image at the start of the article.

Iteration in HTML templates

Our video site now has the ability to play videos stored on the server – but users have no way of knowing which videos are available. In this section, we'll create a home page that lists all the available videos on the site.

First, we'll write a function that will get us a list of IDs for all the videos living on the server. We'll do this by reading the contents of the `videos` directory and putting all the names of the subdirectories into a list. An example implementation which you can use is shown in **Figure 5**. The only new things are the use of the `ioutil.ReadDir` function for reading the contents of a directory, the `for` loop syntax, and the functions `make` and `append`, which we can use for array manipulations. The call `make([]string, 0)` creates a `0`-length array of strings, while the `append` function adds an element to the end of an existing array, resizing the array as required.

Next, we'll create an HTML template for the home page, which we'll save as `home.html` in the `lutube` directory. The template is shown in **Figure 6**. In order to create this template, we've used a range, which enables us to iterate over an array. This template expects to receive an array of `string`s, which it refers to as `{{.}}`. The block starting with `{{range $a = .}}` and ending with `{{end}}` will repeat once for every element of the array; inside the block, we can refer to that element as `{{$a}}`.

> "It's always better to deal with errors early on so that you have a better idea what's happened when things go wrong"

Now we're going to write a `homeHandler` function that will handle requests to the base URL. This function is so similar to the `watchHandler` that we're going to leave it up to you to write it (you can check your work against the version that we'll supply next issue).

Your implementation should call the `getAvailableVideos()`function to get a list of available videos, and should then parse and render the template `home.html`. You should pass the list returned by

Figure 7

```
func renderTemplate(writer http.ResponseWriter, templateFile string, data interface{}) {
    templ, _ := template.ParseFiles(templateFile + ".html")
    templ.Execute(writer, data)
}
func watchHandler(writer http.ResponseWriter, request *http.Request) {
    id := request.URL.Path[len("/watch/"):]
    video, _ := loadVideo(id)
    renderTemplate(writer, "watch", video)
}
```

Top Right In Go, a type implements an interface if it has all the right methods. Therefore, every type is considered to implement the empty interface

Figure 6

```
<h1>LU-Tube</h1>
<p>The video-sharing site for Linux Users</p>

<h2>Our videos</h2>
<ul>
  {{range $a := .}}
  <li><a href="/watch/{{$a}}">{{$a}}</a></li>
  {{end}}
</ul>
```

Right HTML template iteration. Go's HTML template package provides a number of other useful features besides this one for programmatically building web pages in an elegant way

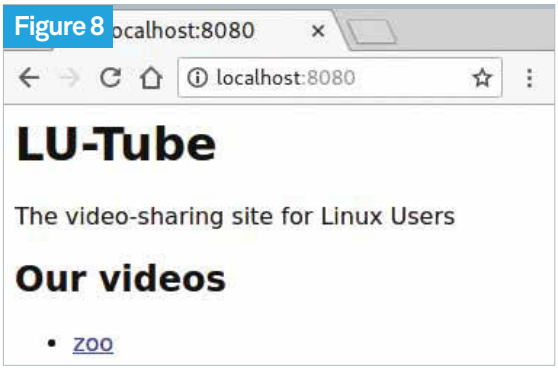
Function literals and closures in Go

Like other languages (even C), Go allows us to pass functions around as arguments to other functions. The way functions are declared means that the syntax is often easier to understand than it is in C, where such declarations are notoriously difficult to understand. For example, the equivalent of the C declaration `int f(int (*g)(char c), char *p)` in Go is `func f(func g(c byte) uint8, p *byte) uint8`. It's easier to see what's going on in the Go version, and becomes easier for more complicated examples.

Go also enables us to use anonymous function literals, in much the same way as JavaScript. For example, we could call the above function `f` as `f(func (c byte) { return 6 }, nil)`.

Function-passing makes error-handling code less repetitive. For example, our handler functions involve much the same error-handling code, but it is difficult to separate out the error code from the code used for handling the requests. The solution is to use function passing: we write a separate function that takes in the request-handling code as a function parameter, and then calls it, performing the appropriate error-handling. We can then wrap our functions inside this new function, meaning that we only have to write the error-handling code once.

Figure 8



`getAvailableVideos()` into the `templ.Execute` function so that it gets used to populate the list of available videos on the page.

You might notice that there is some repeated code between the `watchHandler` and `homeHandler` functions, specifically relating to the handling of the templates. If that is the case, you might like to write a helper function that you can call from each of them.

Figure 7 shows how that might look for the `watchHandler` function. The type `interface {}` is the empty interface: in Go, every object matches this type, so we can pass different types of data into the template rendering functions.

Last, we add a line of code to the `main` function to register our new handler:

```
http.HandleFunc("/", homeHandler)
```

Compile and run the server again, and then navigate to **http://localhost:8080** in your browser. You should see something like **Figure 8**.

Handling errors

Go has no special error-handling functionality beyond the `error` type and the ability of functions to return multiple values, which means that they can return an error value alongside their usual return value(s). We've already seen a few examples of error handling, but we've also been ignoring a lot of the errors in our code. This can cause some confusing behaviour; it's always better to deal with errors early on so that you have a better idea what's happened when things go wrong.

As an example, we'll deal with the errors in the `watchHandler` function. Inside that function, we call the `loadVideo` function, but then throw away the error value:

```
video, _ := loadVideo(id)
```

The only reason why `loadVideo` might fail is if the particular video ID does not exist. You can see this in action by running the server and navigating to **http://localhost:8080/watch/notthere**: since we are ignoring the error, we get a blank page, which isn't particularly helpful. A better thing to do would be to redirect the user to the home page if they accidentally type in an incorrect link, by replacing the line above with the following lines.

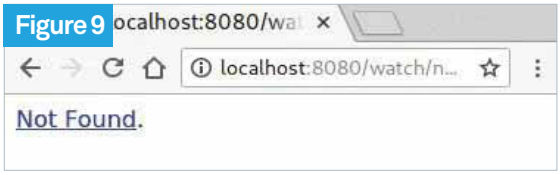
```
video, err := loadVideo(id)
if err != nil {
    http.Redirect(writer, request, "/", http.StatusNotFound)
}
```

If we compile and run the server again, we'll get a link saying 'Not found.' that takes us back to the home page, as in **Figure 9**. If you change the HTTP status code, you can modify the behaviour of the redirect; for example, if we replace `http.StatusNotFound` with `http.StatusSeeOther`, then the page will redirect straight back to the home page; we prefer this approach, since it gives a clear message to the user about what's happened. If you want a challenge, you could try to have an explanatory message on the home page that displays only if a redirect has just occurred, to make things even clearer for the user.

For now, fix up the rest of the error handling, and maybe modify the HTML templates to add some styling and other bits and pieces to make the pages look better (that is, less like a site from 1998). Then get ready to join us next issue when we'll be adding search functionality to our site.

Left A basic home page with a list of all the available videos. We'll leave it up to you to add CSS and other features to make it look more sophisticated

Figure 9

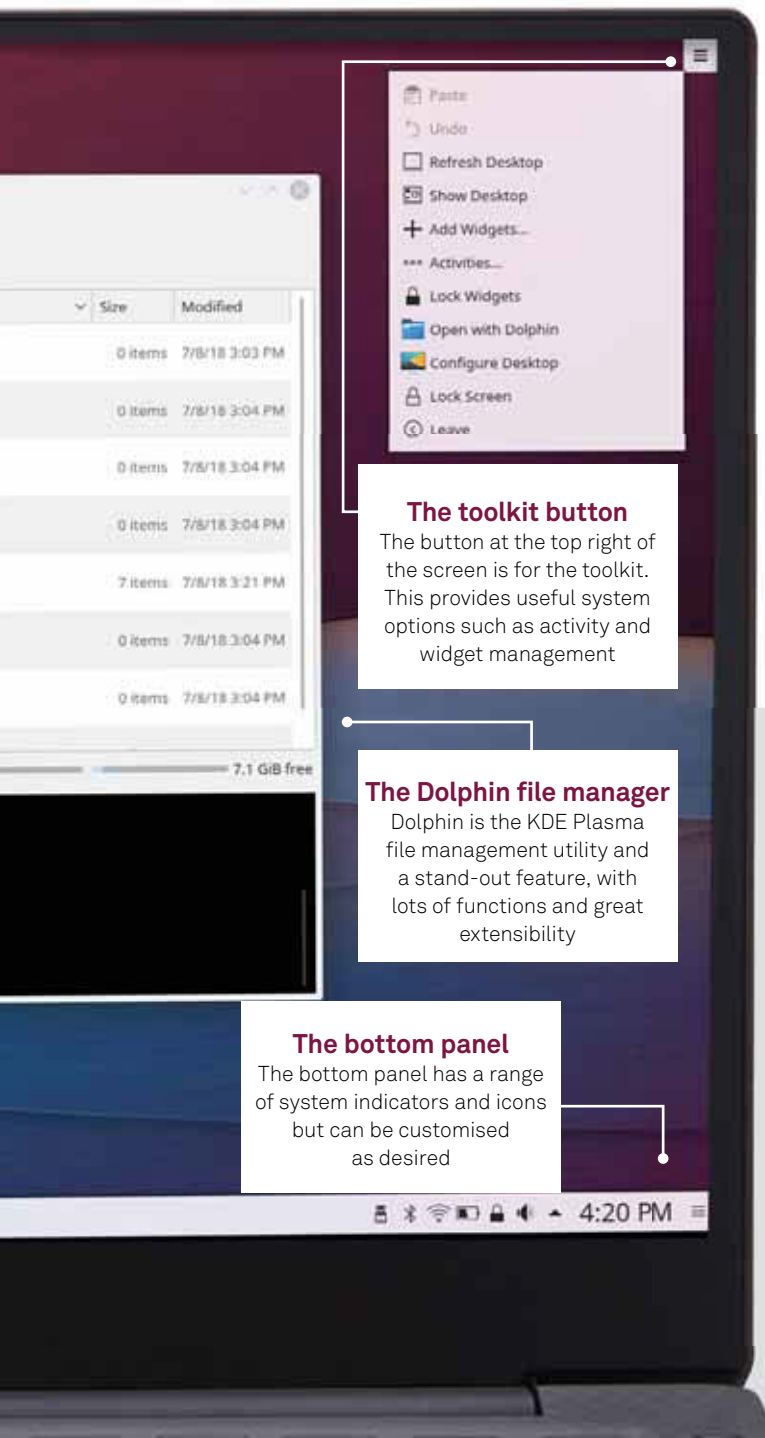


Left The behaviour of a redirect depends on the HTTP status code passed in. To get an automatic redirect, you'll need to use a code in the 3xx range

POWER USER'S GUIDE TO

PLASMA

KDE 5.13


Paste
Undo
Refresh Desktop
Show Desktop
Add Widgets...
Activities...
Lock Widgets
Open with Dolphin
Configure Desktop
Lock Screen
Leave
Size
Modified
7.1 GiB free
4:20 PM
The toolkit button
The button at the top right of the screen is for the toolkit. This provides useful system options such as activity and widget management
The Dolphin file manager
Dolphin is the KDE Plasma file management utility and a stand-out feature, with lots of functions and great extensibility
The bottom panel
The bottom panel has a range of system indicators and icons but can be customised as desired

Paul O'Brien explores the latest KDE Plasma desktop environment, which is more mature than ever before

KDE Plasma 5.13 is the 14th iteration (since the original version was 5.0) of the fifth generation of the desktop environment created by KDE. KDE Plasma sits alongside its fellow KDE Frameworks and KDE Applications projects and is subject to its own release schedule, which typically means main releases every three months, bug-fix releases in between and periodic LTS releases (most recently version 5.12). Initially released on 15 July 2014, KDE is built using Qt5 and KDE Frameworks 5, using OpenGL to enable hardware acceleration for excellent graphical performance.

> "Extensive HiDPI support is included, making KDE Plasma one of the leading desktop environments when it comes to supporting high resolutions"

Extensive HiDPI support is included, making KDE Plasma one of the leading desktop environments when it comes to supporting high resolution and multiple displays – cemented by a new 5.13 feature whereby a dialog now appears as soon as an external monitor is connected, prompting the user to configure HiDPI and position settings. Key components of KDE Plasma include Krunner, the launcher; activities, a powerful expansion of the virtual desktop concept; Plasmoids widgets; and Dolphin, the file manager.

The Plasma team has spent a lot of time working to ensure that not only does the desktop environment maintain a level of polish and a feature set that is rarely matched in the Linux world, it also loads and runs quickly. This is key, as 'lighter' environments continue to be popular and to help Plasma shake a reputation for being power-

hungry and only suitable for use on modern hardware. The performance changes in the new release include optimised start-up times and minimised memory usage, which mean a noticeably faster time-to-desktop over 5.12 LTS, better runtime performance and considerably reduced memory consumption. Basic features such as panel pop-ups have been optimised to ensure they run smoothly even on the lowest-end hardware. The overall look and feel hasn't changed much – Plasma still retains its traditional desktop layout and impressive customisability without being overwhelming. But improvements to overall fit and finish and the Kwin compositor – equivalent to Compiz /Mutter in other distros – which provide clever blur and transition effects

> "Basic features such as panel pop-ups have been optimised to ensure they run smoothly"

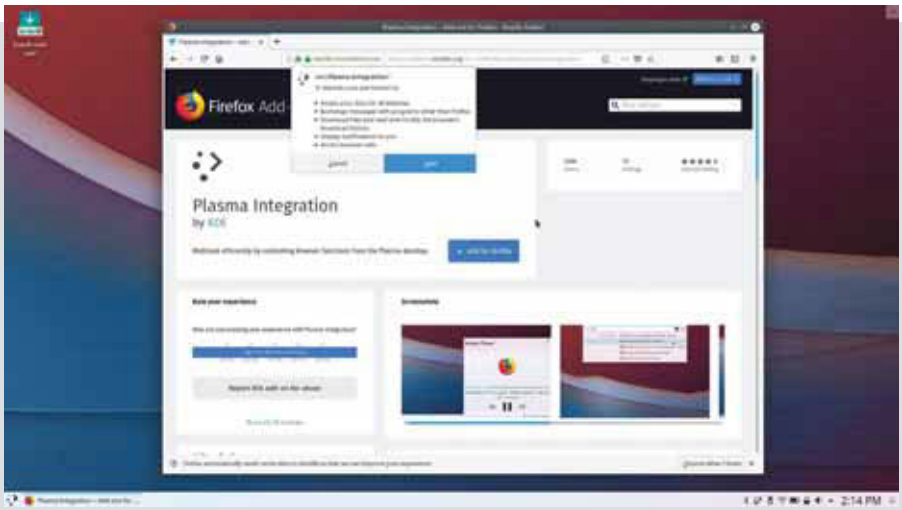
ensure Plasma stays at the leading edge of good-looking desktop environments, and remains more than capable of challenging the best Apple and Microsoft have to offer in terms of visuals.

A key new feature for version 5.13 is 'Plasma Browser Integration', a suite of new features which make Mozilla Firefox, Google Chrome and Chromium-based browsers work more closely with the Plasma desktop. The changes make using Plasma and the internet feel much more cohesive than ever before. Download progress is now displayed in the Plasma notification pop-up just like transferring files with Dolphin, and a native notification is displayed when the download completes. The new 'Clear History' button in the notification widget now becomes particularly useful.

The Media Controls Plasmoid widget, itself redesigned for the latest release, can now mute, pause and skip videos and music playing from within the browser. Other improvements include – courtesy of KDE Connect – the ability to send a link from your browser directly to your Android phone or tablet, and browser tabs can be opened directly from KRunner via the **Alt**+**Space** keyboard shortcut. Enabling Plasma Browser Integration is straightforward, though not configured by default: simply add the relevant plug-in from the add-on store of your preferred browser.

KDE Connect is a particularly useful and often overlooked KDE feature which becomes ever more useful with each release. Connect is a project to enable communication across all your devices. For example, with KDE Connect you can receive phone notifications on your desktop computer, transfer files between your desktop and your phone, quickly find your phone, control music playing on your desktop from your phone, use your phone as a remote control for your desktop, or even synchronise your device clipboards! This is achieved via a component installed on your desktop and a KDE Client app installed on your phone, with the two devices being connected to the same network. Only Android is currently supported.

The Settings pages have started to receive a long overdue redesign for the latest release. The KDE Visual Design Group has,



Above An extension needs to be installed from your browser's store to enable integration features

QUICK GUIDE
Five of our favourite Plasmoids widgets

Plasmoids (aka Plasma 5 Widgets) provide a powerful way to expand KDE Plasma. These are our favourites:

1 **AlphaBlack Control** is a useful widget to control the desktop theme's accent colour, panel opacity, and the style of the taskbar's window buttons.

2 **Event Calendar** is based on the Event Flow Calendar for Android, with daily weather forecasts and the ability to display events from Google Calendar. It also includes a timer and 24-hour weather forecast graph.

3 **Redshift Control** is a tool for enabling and disabling the redshift process, which adjusts the colour temperature of your screen according to environment. It also enables you to access the advanced settings and provides manual mouse-wheel screen-temperature control.

4 **Simple Menu** is an intentionally minimal launcher widget powered by the same powerful back-end as Plasma 5's bundled launchers. Its UI is similar to the full-screen Application Dashboard, but appears in a small, floating window. In addition to the application grid and filtering, it supports favourites, search and basic system actions.

5 **Weather Widget** shows weather information from yr.no and Open Weather Map servers. Features include basic data using font-like icons in a panel, meteogram and long-term forecasts in an expanded Plasmoid, near-future forecasts, plus an offline cache, support for multiple locations.

Above The settings screen gets a Kirigami based make-over to improve its usability

uh, retooled many of the tools in System Settings and their improvements – together with the use of KDE's Kirigami lightweight user-interface framework – give the pages a slick new look that adheres to the Kirigami Human Interface Guidelines. As a bonus they're also more touch-friendly than before, as well as revealing new functionality. The team has started a rolling programme of improvements to the theming tools section, which includes icons, desktop themes and cursor themes. The splash screen page can now download new splash screens from the KDE Store, and the fonts page can now display previews for sub-pixel anti-aliasing.

Additional visual improvements come courtesy of a new design for the lock, unlock and login screens, which now display the wallpaper of the current Plasma release by default. This makes the screens feel much more integrated with the rest of the system compared to previous iterations. The lock screen also includes a fade-to-blur transition to show the controls, allowing it to more easily be used like a screensaver.

Discover, the KDE Plasma software and add-on installer, receives additional features and design improvements for the 5.13 release. Once again, the Kirigami UI framework is put to good use and the team has improved the appearance of lists and category pages, which now use toolbars instead of big banner images. Lists can now be sorted and use the new Kirigami Cards widget. Star ratings are shown on lists and app pages to more easily identify high-quality items, and the overall store-browsing experience is considerably improved. All Upstream data (the cross-distro XML format to provide metadata for software components) is now shown on the application page, including all URL types, while app icons use the local icon theme to better match your desktop settings.

> "The team has started a rolling programme of improvements to the theming tools"

Under the covers, work has continued on bundled app formats. Enhanced Snap support now enables users to control app permissions and it's possible to install Snaps that use classic mode. Deep linking of Snap packages is now supported courtesy of the **snap://** URL format. Flatpak hasn't been left behind, with an added ability to choose the preferred repository to install from when more than one is configured.

QUICK TIP

Dolphin add-ons

The available Dolphin add-ons enable a wide range of features to be added to the standard file manager setup, with nearly 500 options available in the store! Dropbox integration is a firm favourite for teams, as are a range of root utilities.

TIPS & TRICKS

Be a power user

How to make the most of your KDE Plasma experience

1 Install Plasma add-ons
The Software Centre has an entire section dedicated to Plasma add-ons, sorted by category, so make the most of it! Window Manager scripts provide useful inspiration for how to use Window Rules, and Plasma widgets are perfect for customising your various activities.

2 Make the most of KRunner
There's no need to manually dive through the application menu every time you want to launch an application. Provided the desktop has focus or after pressing the super key, just start typing the name of the app you want to open and hit Return when the list is suitably refined.

3 Use jump lists
Jump lists enable you to launch an application directly to a specific section or feature. You can see this in action by adding the Software Centre to a panel, then right-clicking the icon. You'll see an option to jump directly to updates. Application support varies of course, but it is useful.

4 Be a Dolphin power user
Dolphin is a fully featured and accomplished file manager, despite appearing simple at first glance. Check out the Preview option to get a look at your files, the Split feature to make file management operations far easier, and root around in the Control menu to find useful options to match your preferences.

5 Monitor services
KDE Plasma includes a useful utility for understanding which services are running in the background and which start with your computer. Available in the 'Start and Shutdown' section of System Settings, the module allows services to be manually stopped and started.

One feature not quite ready for prime-time in this release, but included as a technology preview, is GTK global menu integration. The feature, particularly useful for users that rely on GTK apps such as GIMP, LibreOffice and the GNOME suite of apps, is enabled courtesy of a proxy package that interfaces between the two GMenu and DBusMenu protocols. The solution means that GTK apps will gain global menu support with no modifications needed. Despite still being a tech preview, most apps already work very well, with only a handful of those we tried not working and no general ill effects of having the proxy installed.

> “Plasma Vault enables you to create encrypted folders to securely store files”

Plasma Vault has several new enhancements for Plasma 5.13. Plasma Vault enables a user to create encrypted folders to securely store files. As the developer of Plasma Vault Ivan Cukic notes, the utility covers a different use case to full-disk encryption or an encrypted home folder. “[They do] not cover the possibility that someone might access your system while it is running. Plasma Vaults fill this void by making the attack surface smaller – instead of having all data unlocked at once, you can do it piece by piece – it is more granular.” The latest release introduces a new CryFS back-end, commands to remotely close open vaults from your Android smartphone or tablet with KDE Connect, offline vaults, a more polished interface and better error reporting.

Above It’s easier than before to find interesting and top-rated software in the KDE software store

A new feature that may sound trivial, but which will make a lot of people’s Plasma lives a little more bearable, is the addition of the ability to fall back to software rendering if the OpenGL drivers unexpectedly fail. While in theory this is a scenario that shouldn’t occur, Nvidia graphics users will know it happens more often than one might like – and the software rendering fall-back will at least make the unfortunate process slightly more graceful.

QUICK TIP

Terminal quick-access

When browsing a directory within the lightweight Dolphin file manager, a great shortcut to know is that you can instantly open a terminal window in the current directory, attached to the Dolphin window, by pressing **F4**.

While these are the main improvements in Plasma 5.13, there are numerous other small tweaks, including a Plasma Calendar plug-in for astronomical events showing lunar phases and astronomical seasons (equinoxes and solstices), an improvement to the Digital Clock widget which enables copying the current date and time to the clipboard, additional KRunner plug-ins to provide easy access to Konsole profiles and the character picker – and the Mouse System Settings page has been rewritten for libinput support on X and Wayland.

Assuming you like what you see and want to try KDE Plasma 5.13 – and, unless you specifically want to stay on the 5.12 LTS release, why wouldn’t you? – there are several ways to get it up and running. The most straightforward is to use a live image, a number of which are available. KDE’s own Neon image builds the latest KDE Frameworks, Plasma and Applications courtesy of a continuous integration system on top of an Ubuntu LTS base. Several versions are available, including the User Edition which features the latest officially released KDE software on a stable base and is ideal for everyday users. The Developer Edition Git-Unstable version features

QUICK GUIDE

Creating and using activities

By default, the first time you fire up KDE Plasma you’ll be greeted with a single activity. The activity switcher won’t be visible, as it only shows when you have more than one active, but if you don’t make the most of Activities, you’re missing out. To get started, press the toolkit button at the top right of the screen and select Activities. You’ll be presented with your current activities (likely only ‘default’) and a ‘Create activity’ button to add more. These are groups of applications and a unique desktop where you can add widgets. You can control how an activity is displayed, with a text name or an icon, a keyboard shortcut can be configured, and activity tracking can be disabled if you don’t want the apps to appear in your launcher history. Activities can even have unique settings for screen brightness, wireless settings and indeed anything else via custom scripts – ideal if you want to configure a ‘presentation mode’ or similar. Activities can also have their own desktop settings such as wallpaper, mouse actions, icon arrangement or even file filtering.

pre-release KDE software built the same day from new feature branches, while the Developer Edition Git-Stable version has pre-release KDE software built the same day from bug-fix branches. Outside of the official KDE images, the latest released Plasma version is also available in the openSUSE Tumbleweed-based Krypton stable image, both in 32-bit and 64-bit flavours (bearing in mind that Neon itself is only available in 64-bit form).

GeckoLinux, which focuses on out-of-the-box usability, is built directly from openSUSE packages and repositories, offers the Plasma 5.13 desktop environment on top of the Leap 15.0 base system in its NEXT image. It also includes KDE Applications 18.04 and the KDE Framework at version 5.47.0. The GeckoLinux ROLLING Plasma can also be updated to Plasma 5.13.

> "Neon builds the latest KDE Frameworks, Plasma and Applications on top of an Ubuntu LTS base"

Manjaro KDE Edition is a rolling-release distribution, which provides the latest stable Plasma, KDE Applications, KDE Frameworks and Qt releases. Should Slackware be your preferred distro, Alien BOB provides an installable Live image for Plasma 5, based on the development tree 'slackware64-current'.

If you're a Kubuntu 18.04 LTS user, or using another Ubuntu distro with KDE installed, you can now install Plasma 5.13 by adding the Kubuntu Backports PPA to your software sources. Use the commands `sudo add-apt-repository ppa:kubuntu-ppa/backports` and `sudo apt update && sudo apt full-upgrade` to add the repo and install, but remember that packages in the backports PPA have reduced level of support compared to the main repos and may introduce bugs.

The move towards Wayland from X.Org server is now underway in a number of projects, and KDE Plasma is no exception. After Plasma 5.12 was released in February, the team got together to discuss improvements to Wayland support for the 5.13 release. Support in 5.13 is still considered very early, with improvements including the return of Window Rules, the use of high-priority EGL Contexts and initial support for screencasts and desktop sharing; but work on Wayland is expected to accelerate over the coming months. The KWin/X11 code is now considered feature-frozen, as confirmed by KWin maintainer Martin Flöser, which will enables the teams to focus on support. This doesn't mean that 5.14 will default to a Wayland-based session – that still feels some time off – but things are definitely moving in the right direction. ■

QUICK GUIDE

Using Window Rules

One of the best things about KDE Plasma is that it's instantly accessible, yet has a huge number of powerful features. Window Rules are a great example of this: not something you'd immediately think of as being useful, but a must-have when you get used to having them! Window Rules, also known as Window-Specific settings, can be access from the 'Window Management' section of the revamped System Settings application. They enable you to select a window using a whole host of options including application, role, type, title or even machine, then apply a vast number of custom attributes including size and position, activity, maximum or minimum size and priority above other windows. The title bar and frame can be disabled, the colour scheme changed, and opacity applied. Windows can be set as uncloseable, unfocusable and much more, providing ultimate versatility to how windows behave on your system. Like to watch video? You could remove the title bar and frame from the player, make it semi-opaque, pin it on top of every other window and carry on with your work.

QUICK TIP

Add your online accounts

Online accounts can be added to Plasma to enable automatic integration with key applications. You may need to manually install the 'kaccounts-integration' package to get started, then select 'Online Accounts' in system settings.

Right The new Plasma lock screen features updated wallpaper support and fade-to-blur transitions to reveal controls, which means that it works well as a screensaver too. The login screen gets the same lick of paint

COMPUTING WITH A DOUBLE HELIX

The molecule of life is being pressed into service for computing with some impressive and surprising results. **Mike Bedford** reveals how

AT A GLANCE

- DNA roots **p64**
  Early DNA research enabled massively parallel computation for niche applications.
- DNA logic **p65**
  Logic gates can be made from DNA, but they won't get any prizes for speed.
- Non-deterministic DNA **p67**
  A non-deterministic architecture offers the promise of universality and parallelism.
- DNA storage **p68**
  DNA-based storage could offer data retention that is measured in the hundreds of years.

The silicon era has had a good innings. Since the dawn of silicon chips, we've seen memory increase in capacity by a factor of 128 million and processors become several billion times faster. Yet the writing is on the wall as semiconductor manufacturers now struggle to produce the ever smaller feature sizes needed to give us the year-on-year improvements we've grown to expect. It might be bit premature to sound the death knell for silicon anytime soon, and history suggests that most such problems have been solved sooner or later, but some scientists are taking no chances.

So while electronics engineers are continuing to extract yet more performance from conventional silicon circuits, other researchers have their sights set on a whole host of revolutionary and unexpected technologies. Here we check out one potential new technology: computing with DNA.

Although we've become accustomed to computing being first and foremost about electronics, in one sense we shouldn't be too surprised that DNA is a potential successor. After all, this double helix molecule contains all the data necessary to replicate a human being and, in that sense, is a form of data storage. When we bear in mind that the development of the electronic stored program computer relied on the development of electronic memory, we can speculate that this chemical memory might do the same for DNA computers. Knowing something about the structure of the DNA molecule and how it replicates is key to understanding how it can be used in computing.

DNA roots

Despite its undeniably futuristic nature, DNA computing can trace its ancestry to a 1994 paper by Leonard Adleman of the University of Southern California. The topic he chose was the Travelling Salesman Problem. The task involves deciding if it is possible for a salesman to visit a list of cities in any order, with fixed start and end locations, passing through each city only once given a list of available direct flights between cities. The normal method of solving the problem is basically a trial and error approach but the time taken increases dramatically with the number of cities. Not so with Adleman's computer-in-a-test-tube.

In this pioneering work, just seven cities interconnected by 14 flights were used, but the method has potential to be scaled up. Here, however, we've simplified things further by using just four cities and six flights. The four cities are Atlanta, Boston, Chicago and Detroit, and the available one-way flights are from Atlanta to Boston and Detroit, Boston to Atlanta, Chicago and Detroit, and Chicago to Detroit. The starting city is Atlanta and the final city is Detroit.

The method involves representing the cities and flights by 8-base DNA strands according to strict

rules. The sequence for the DNA strand representing a flight is the complement of the final four bases of the 'from' city, followed by the complement of the first four bases of the 'to' city. The top part of the diagram on p66 shows the strands for Atlanta and Boston bonded to the strand for the Atlanta-to-Boston flight, the only flight they will jointly bond with (with the double-strand helix straightened out for clarity).

Calculating the solution involves preparing samples of DNA for the cities and flights and mixing them together a pinch of each so that chemical reactions can take place. Even such a small amount of each type of DNA contains about a hundred trillion molecules which, almost certainly, would result in all possible combinations of cities and flights being generated by chemical reactions. While all those combinations will be legal in the sense that 'A's would only bond to 'T's and 'C's to 'G's, not all would be valid solutions to the problem. For example, some wouldn't start at Atlanta and end at Detroit and others would miss out a city.

The bottom part of the diagram shows the only valid solution and a couple of invalid ones. Although the correct answer was generated in a few seconds, it was then necessary to isolate it from all the invalid ones. This job took about a week of supplementary chemical reactions, making the whole task much more long-winded that it would have been on a PC, but this was, nevertheless, the birth of a new era of computing. What's more, it served to highlight DNA's potential for massively parallel computation.

> "DNA computing can trace its ancestry to a 1994 paper by Leonard Adleman of the University of Southern California"

DNA logic

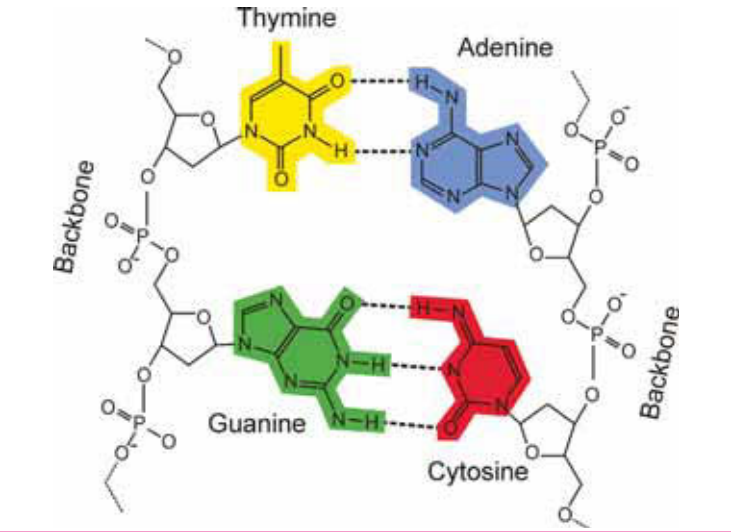
Impressive as it might be to have hundreds of trillions of computations take place in a few seconds, Leonard Adleman's historic DNA computing architecture was hard-wired to solve a particular problem. By way of contrast, we've grown accustomed to general-purpose computing hardware that carries out a particular task according to the program stored in its memory. Because the basic building blocks of logic gates are at the heart of general-purpose electronic computers, several researchers have turned their attention to implementing logic gates using DNA.

Most DNA logic gates rely on a chemical reaction known as toehold moderated strand displacement. While reading the following description, we suggest that you refer to the top part of the diagram on p67.

QUICK GUIDE

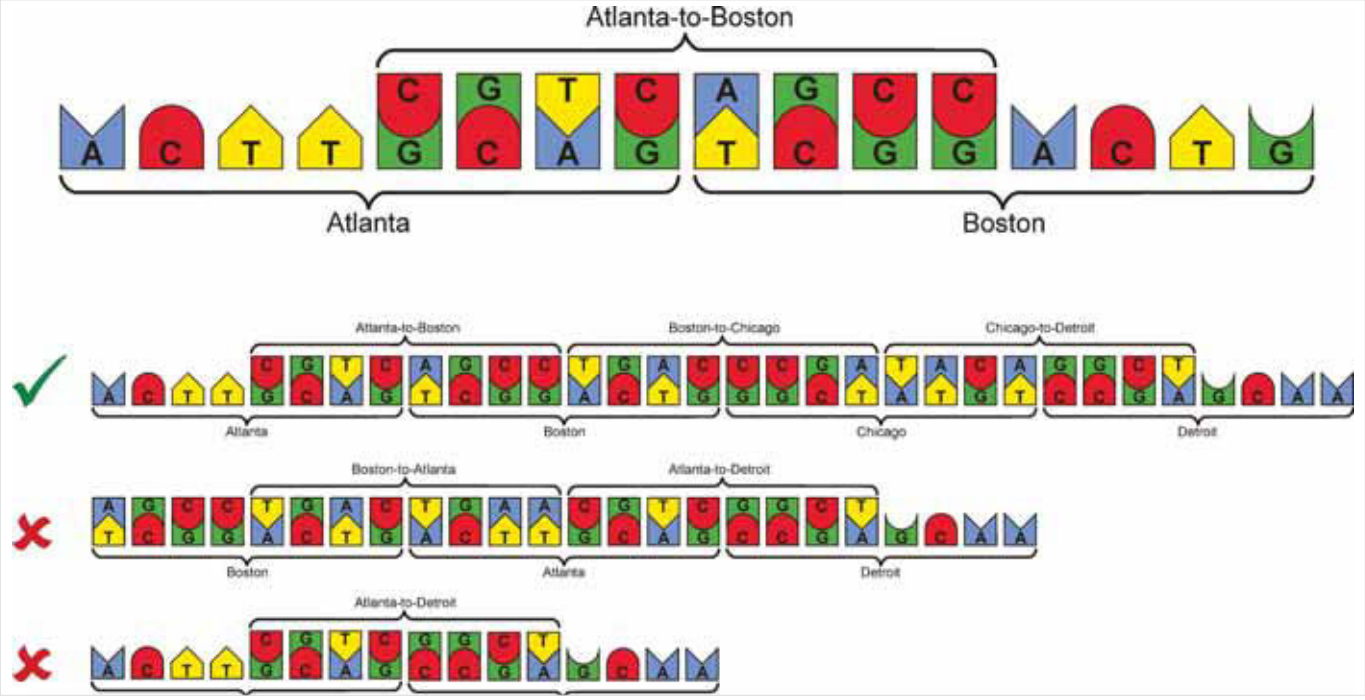
DNA structure

Each strand of the double helix of a DNA module is formed of molecular groups called nucleotides. A nucleotide is a combination of three smaller types of group: a sugar, one or more phosphate groups and a nitrogenous base. There are four types of bases called Adenine (A), Thymine (T), Guanine (G) and Cytosine (C), and the nucleotides containing these bases can appear in any order along a strand of DNA. While the nucleotides are bonded together with strong chemical bonds in a strand, the bases can also bond to other bases via a weaker so-called hydrogen bond, linking the two stands. These wind around each other to form a double helix. A will only bond with its complementary base T, and C will only bond with G – so for every A in one strand there is an adjacent C in the other strand. Shown here is a segment of DNA, straightened for clarity.



The reaction is between two DNA molecules. One is a single strand and the other a double strand but with the strands offset to leave some single sections, the one coloured light brown being the toehold. Unlike the diagrams in our Quick Guides to DNA structure and DNA replication, we are using colours not to represent nucleotides but DNA strands. What's more, colours now represent a unique strand (a unique sequence of nucleotides), while a dark and a light strand of the same colour represent complementary strands. By that we mean strands that have As and Ts interchanged and Cs and Gs interchanged so that they will bond to form a double strand.

You'll notice that the single DNA strand in part A has the complementary strand to the toehold part of the double strand – the dark brown section – so those two portions of the two molecules will bond together. This gives the single strand a toehold into the double strand, as shown in part B, and this is the start of a fuller reaction. This is shown in part C where the rest of the single strand bonds with the double strand,



Above DNA computation has been used to solve the computationally intensive Travelling Salesman Problem using a massively parallel architecture

displacing the original shorter part of the double strand in so doing. We can now build on this basic mechanism to see how this reaction could be used to implement an OR gate, one of the logic gates that form universal electronic computers. This isn't necessarily a practical OR gate and, at the very least, it is a gross over-simplification, but we trust that it provides an inkling of how DNA can be used to implement logic circuitry. In the following description, we are referring to the bottom part of the same diagram.

You'll notice that the single DNA strands are called inputs and outputs, while the double strands are the gates. These are not directly equivalent to logic gates, though; in fact, three such gates are needed to construct a single OR gate. The other thing to point out is that, unlike the case with electronic circuits where 0s and 1s are represented by different voltages, 0s and 1s are represented by the absence or presence of a particular single strand of DNA. In part D we can see the OR gate, comprising three DNA gates; with no inputs, no reactions take place so there is no output. Inputs of 0 and 0, therefore, give an output of 0, which is what would be expected of an OR gate. In part E we introduce an input and it's clear that this will react with Gate 1 to generate an output. This is what has happened in part F, and we can also see that the output from Gate 1 will react with Gate 3. This has happened in part G, the single strand being the output of the OR gate.

> "DNA gates aren't hard-wired to each other; instead, the gates and the input and output strands are free to move in solution"

Inputs of 1 and 0, therefore, have generated an output of 1 and, again, this is what is expected of an OR gate. It is fairly clear that a different input (such as anything:brown:orange) would react with Gate 2 to produce an output, that would in turn react with Gate 3 to generate the same output – and both inputs together would also generate an output.

The above description raises a couple of questions. First, how is the output from one gate routed to the input of another and, second, why is the green:brown:black strand considered an output of the OR gate as a whole, while the intermediate purple:brown:green strand is not?

In fact, the answers to both questions are related. DNA gates aren't hard-wired to each other as the gates on a silicon chip are. Instead, the gates and the input and output strands are free to move in solution and will do so by diffusion. A single input strand will, therefore, come into contact with lots of gates, but will only react with the ones with a complementary strand.

In our example, therefore, Input 1 could well come into contact with Gate 3, but it won't react since it was designed to react only with Gate 1. In other words, different gates have different input and output strands, even though they all represent a binary 1. This also explains why the presence of the purple:brown:green strand isn't considered to be an output of the complete OR gate. In any practical circuit, the output of the OR gate will be routed to the input of another logic gate, but that logic gate will be

QUICK TIP

Ultra-low power

Supercomputers are hugely power hungry but one based on DNA, like the University of Manchester's proposed machine, could be much more frugal. Researchers say it could perform 2 x 10^{19} operations per joule – that's a billion times more than an electronic equivalent.

designed to respond to the green:brown:black strand and not purple:brown:green.

Caltech researchers Erik Winfree and Lulu Qian have already created circuits made from multiple DNA logic gates to perform arithmetic calculations. Their circuit used 130 strands of DNA and was able to calculate the square root of a 4-bit binary number in 6 to 10 hours. More recently, researchers at the University of Washington have used so-called DNA origami to act as scaffolding to anchor the various DNA strands in close proximity to the strands they are designed to react with. This very much reduces the time taken for reacting strands to diffuse towards each other, and has reduced times for simple computations from hours to minutes.

The fact is, though, that these chemical reactions are never going to come close to reaching the speed of electronic circuits, and fast computation is only going to be achieved by offsetting slow chemical reactions with a massively parallel architecture. Certainly logic circuits

> “DNA computation offers major benefits even in cases where speed isn’t one of them”

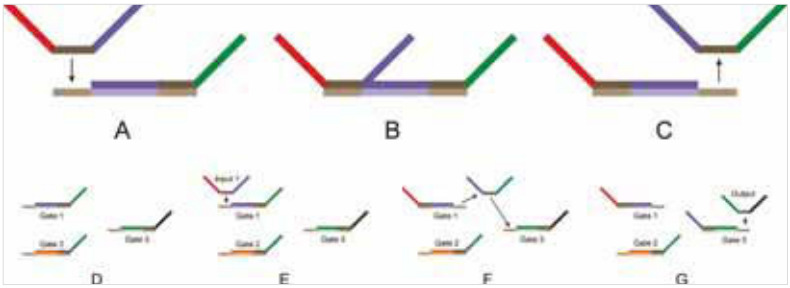
can be designed to exhibit parallelism, but massive gains in performance are only going to be realised when there is the scope for inherent parallelism of the type found in Adleman’s solution to the travelling salesman problem. On the other hand, as many DNA computing pioneers have pointed out, DNA computation offers major benefits even in cases where speed isn’t one of them.

Non-deterministic DNA

So far we’ve seen an example of a massively parallel DNA computation hard-wired to solve one particular problem. We’ve also seen the implementation of logic gates in DNA, with the potential for universal computation, but with much less scope for inherent parallelism. Neither seems to take us close to a super-fast, general purpose DNA computer. However, a 2017 project at the University of Manchester holds out the promise of exactly that.

The development was of a so-called non-deterministic architecture and this is the key to the phenomenal performance on offer. We are used to the idea of computers being deterministic – that is, given a particular program instruction and particular data, the action it will take is fixed. In a non-deterministic architecture, however, there is no such direct link between cause and effect. Instead, it is possible to write a program in which instructions provide several alternative courses of action. Until now, such computers were mere thought experiments and, even if they were physically possible, they probably wouldn’t give a correct answer to a problem because they might guess the wrong course of action from the alternatives provided in the program. The University of Manchester’s novel DNA-based architecture changes all that.

The university’s Professor Ross King likes to give the example of solving a maze to show how this might work. Each time a junction is reached, rather than following one of the two paths at a time as a conventional program would, the code specifies that either path could be taken – and the inherently parallel nature of the architecture enables both to be taken simultaneously. Once these two paths reach further junctions, the computer would find itself following four paths concurrently – and so on, with the number of parallel paths increasing exponentially.



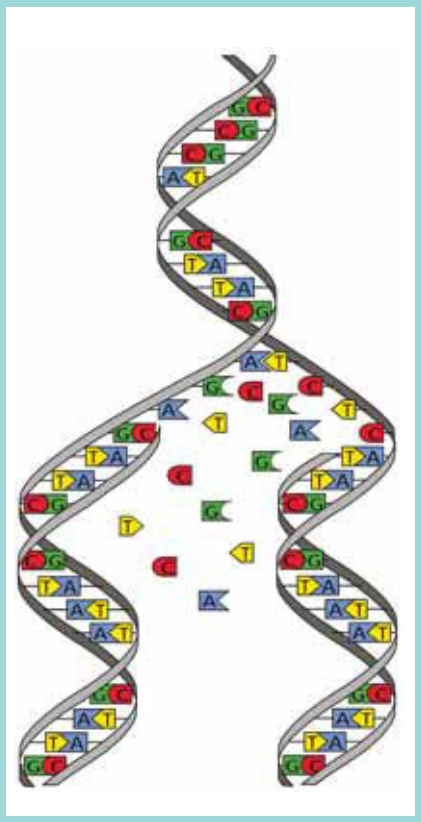
Below The toehold moderated strand displacement enables logic gates to be constructed from DNA

QUICK GUIDE

DNA replication

DNA replicates itself in cell division via the polymerase chain reaction which is also key to some DNA-computing architectures. Replication starts with the double helix starting to unzip itself by breaking the hydrogen bonds that keep the two strands together. As the two strands start to separate, the now un-bonded bases are free to bond to bases in other nucleotides.

So long as there are free nucleotides around, this is exactly what happens – but only nucleotides with the complementary bases can create new hydrogen bonds. So As and Ts attract each other, as do Cs and Gs and, in so doing, both strands of the original DNA bond to the nucleotides necessary to replace the missing complementary strand. In time, this process produces two DNA molecules identical to the original one.



QUICK TIP

Time vs. mass

DNA technology could reduce processing time overheads, but at a different sort of cost. Tiny as DNA molecules are, repeatedly doubling their number, as is required for some tasks, means you'd eventually need more DNA than the mass of the earth.

Formerly a pipe dream, DNA makes this feasible, as King explained. "To achieve this we utilise DNA's ability to replicate. At any choice point in a computation, our NUTM [Non-deterministic Universal Turing Machine] replicates itself and takes all possible choices."

This DNA research uses Thue strings, which offer a quite different model of computation to the one we're familiar with, as Professor King elaborates. "The Thue string model is a rewriting system. For example, if you see the string 'ac' you can rewrite it as 'ca', and vice-versa. The non-determinism comes from the fact that there may be multiple choices of possible rewrites in a string".

Unfortunately, it appears that programming it is by no means a trivial task. "We know from the theory of computer science that the Thue system is universal – that is, it can be used to compute anything any other computer can do. However, that does not mean that programming it is easy; it is even more difficult than a classical Turing Machine. Indeed, neither I nor my colleagues have coded anything significant. It would be an interesting challenge to your readers to emulate the Thue system and to use it to, say, encode the addition of two numbers".

So what of the future for NUTMs? Will we ever see them on the desktop? The fact is that DNA reactions will never be fast, but this can be more than offset by the massive parallelism on offer. So, for general purpose computing, with minimal scope for high levels of parallelism, simple problems would actually take longer.

In the elite world of high-performance computing, though, things couldn't be more different. We could envisage a scenario in which DNA-based NUTM co-processors work alongside conventional silicon processors in supercomputers to achieve a massive increase in performance and, at the same time, a huge reduction in energy consumption.

DNA storage

While DNA computers are likely to be niche products, there is evidence that DNA as a storage media may soon be much more mainstream. In nature, the sequence of nucleotides in each strand of DNA represents the data that describes the very nature of the species and the traits of the individual. But outside a living being, the sequence of DNA could, in theory, represent anything. If we bear in mind that there are four different bases, each base could potentially represent two bits of information.

Things aren't quite that simple because DNA reactions aren't 100 per cent repeatable and this would lead to corrupted data; they are slow so reading and writing would take place at a snail's pace; and currently costs would be astronomical. But methods have been devised to counter the effects of data loss by using redundancy, prices will certainly reduce if and when the technology is commercialised, and a slow access speed isn't a problem for its intended application.

That application is data archiving, and here the benefits of DNA are its longevity and data density. When we recall that DNA testing was a key element in identifying the 'skeleton in a car park' as the remains of Richard III a few years ago, the fact that DNA can survive for hundreds of years, even in non-ideal conditions, is evident. In fact, DNA has been extracted and sequenced from Egyptian mummies and from extinct animals such as mammoths and sabre-toothed cats, which makes the 10-100 year lifetime of optical discs, magnetic storage and flash memory look rather puny by comparison.

> "Researchers at Microsoft and the University of Washington encoded 215 petabytes into a gram of DNA"

Meanwhile, data density is absolutely staggering. Researchers at Microsoft and the University of Washington have encoded 215 petabytes (215,000 terabytes) into a gram of DNA; at this density, all the world's data could be stored in 10 tonnes of DNA, which could fit into a trailer. By way of comparison, it's been estimated that the same data on a stack of CDs would reach beyond the moon.

Doug Carmean of Microsoft Research is widely reported as saying that they aim to have a "proto-commercial system in three years storing some amount of data on DNA in one of our data centers, for at least a boutique application". Indeed, speculation is rife that Microsoft plans to add DNA storage to its cloud-based services in the not too distant future.

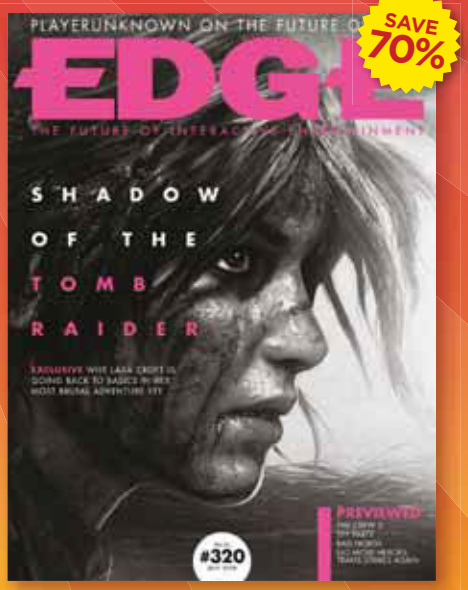
So will we shortly have DNA in our desktop PCs for day-to-day work? Probably not thanks to the speed deficit, but the molecule of life might soon be immortalising our cherished photos and irreplaceable data instead. And let's face it, if we ever do have DNA processors injected into our bloodstream, that will surely be the ultimate in personal computing.

Below DNA storage is already invading popular culture as demonstrated by Netflix's sci-fi TV show on, 3%, which showcased protein-bonds, similar to DNA, being used to create a vast database on the world's population.

SUMMER SALE!

TRY THREE ISSUES FOR £5*

BIG SAVINGS ON OUR BEST-SELLING MAGAZINES



For great savings on all of our magazines, see the entire range online

myfavouritemagazines.co.uk/summer182

Hotline **0344 848 2852**

***TERMS AND CONDITIONS:** The trial offer is for new UK print subscribers paying by Direct Debit only. Savings are compared to buying full priced print issues. You can write to us or call us to cancel your subscription within 14 days of purchase. Payment is non-refundable after the 14 day cancellation period unless exceptional circumstances apply. Your statutory rights are not affected. Prices correct at point of print and subject to change. Full details of the Direct Debit guarantee are available upon request. UK calls will cost the same as other standard fixed line numbers (starting 01 or 02) are included as part of any inclusive or free minutes allowances (if offered by your phone tariff). For full terms and conditions please visit: bit.ly/magtandc. Offer ends 31st July 2018

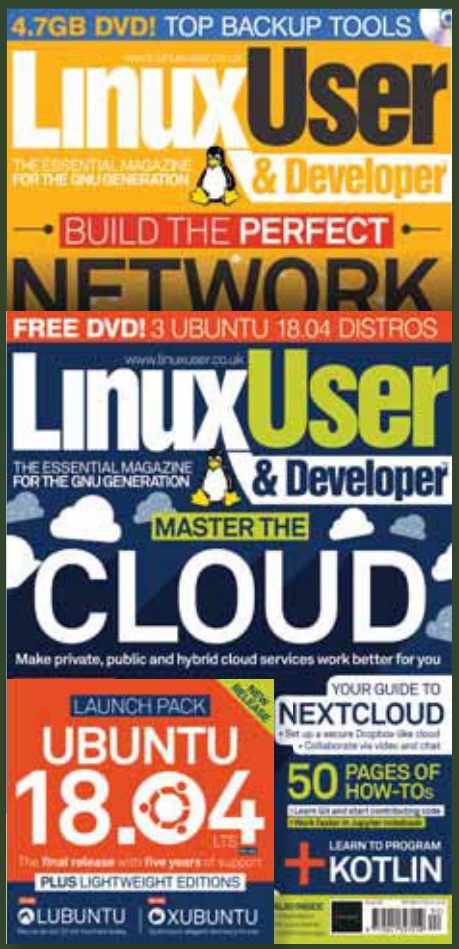
Subscribe | Never miss an issue

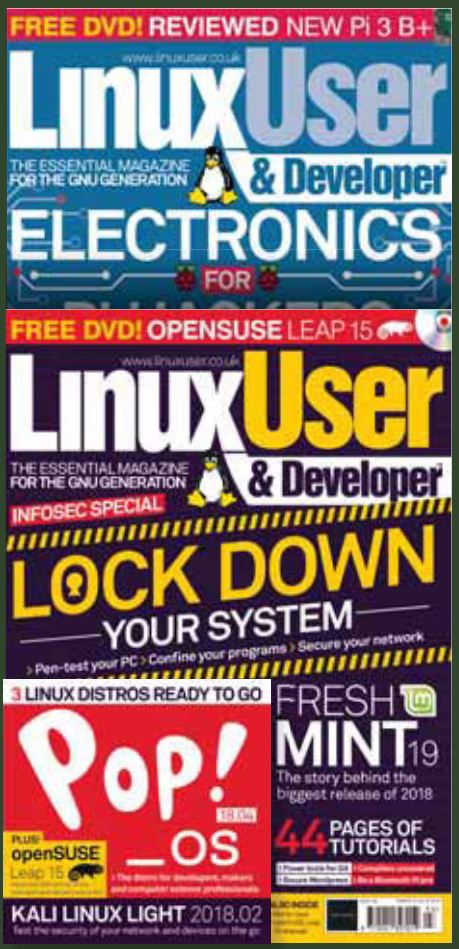
SPECIAL **US** OFFER

OFFER ENDS **SEPTEMBER 30 2018!**

SUBSCRIBE

& GET 6 ISSUES FREE*




FREE DVD! REVIEWED NEW Pi 3 B+
Linux User & Developer
ELECTRONICS
FOR
FREE DVD! OPENSUSE LEAP 15
Linux User & Developer
INFOSEC SPECIAL
LOCK DOWN
YOUR SYSTEM
Pen-test your PC > Confine your programs > Secure your network
3 LINUX DISTROS READY TO GO
Pop! _OS
PLUS openSUSE Leap 15
KALI LINUX LIGHT 2018.02
FRESH MINT 19
The story behind the biggest release of 2018
44 PAGES OF TUTORIALS

FREE RESOURCES! FOR SUBSCRIBERS See p.96

ORDER ONLINE & SAVE

www.myfavouritemagazines.co.uk/sublud

OR CALL 0344 848 2852

* This is a US subscription offer. '6 issues free' refers to the US newsstand price of $16.99 for 13 issues being $220.87, compared to $112.23 for a subscription. You will receive 13 issues in a year. You can write us or call us to cancel your subscription within 14 days of purchase. Payment is non-refundable after the 14 day cancellation period unless exceptional circumstances apply. Your statutory rights are not affected. Prices correct at point of print and subject to change. Full details of the Direct Debit guarantee are available upon request. UK calls cost the same as other standard fixed line numbers (starting 01 or 02) included as part of any inclusive or free minutes allowances (if offered by your phone tariff). For full terms and conditions please visit **bit.ly/magtandc. Offer ends September 30 2018.**

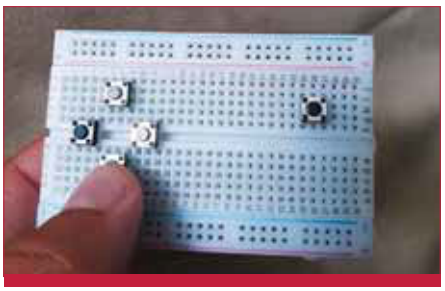
THE ESSENTIAL GUIDE FOR **CODERS & MAKERS**

PRACTICAL Raspberry Pi

72 "I decided to use MQTT as it was simple to set up"

Contents

72 A cool laser-tag game made with a Pi and 3D-printed guns



74 Create a gaming controller prototype for the Pi

76 Get started with the Pimoroni Inky pHAT E-ink display

Pi Laser Tag

Pew pew pew! A Utah State student creates a Pi-powered shooting game with potential for future enhancements

Terran Gerratt

Terran is an electrical engineering student at Utah State University and is employed at WiTricity. He likes wakeboarding with his wife.

Like it?

To dive into more detail of Terran's project head to **http://bit.ly/PiLaserTag**.

Further reading

This is Terran's first public project, follow him on Instructables or LinkedIn (**http://bit.ly/TerranGerratt**). Terran says his next project will be to create "a distributed system robot to accomplish mundane tasks around the house, such as automated blinds or light switches."

Terran's first publicly released project uses a server – essentially any old desktop you might have spare – and Raspberry Pi-powered guns to create a fun infrared laser tag game for three players. Terran says the project relies heavily on Wi-Fi, which is why he decided to use the Raspberry Pi Zero W for the 3D-printed guns.

What inspired you to make your own laser tag game?
During the past year or two, I have been more and more interested in home-automation projects. I first created a simple infrared-relay module using an Arduino to control a fan in my house. I then began realising how easy it was to send information through infrared and wanted to take it to the next level. If I could turn on a fan using a dinky remote, then surely I could shoot it from a 3D-printed gun?

What made you decide to use MQTT? Were there other options you considered?
I decided to use MQTT as it was simple to set up and embed in the code already running on each of the devices. It was also a great place to start, as there are numerous resources and examples online that helped me mould the messaging service into a network for keeping track of game statistics. Looking back on the project and realising how I ended up using the messages, using ROS would be an alternative to MQTT as it has a lot of the needed features prepackaged and ready to go.

Can you give us an overview of how the server and guns communicate with each other?
The project actually used two methods of communication. The first was through infrared codes, as the guns needed to 'shoot' each other, while all of the actual game functions and replies were handled using MQTT. At the beginning of each game, a few game preferences could be chosen on the server and then sent to each player.

Once all of the players were logged into the player lobby, the server started the game. Each Pi was assigned a unique infrared signal so the receiver would be able to distinguish which player had sent the infrared code. The tagged player would then send a reply to the tagger to notify that he was tagged. A copy of every message sent between the players was sent to the server, which compiled all of the messages into a report that was displayed at the end of each game.

You mention using Pygubu for creating the UI?
Yes. As this was my first attempt of creating a python GUI, there was definitely a learning curve that had to be overcome. Using Pygubu to aid in that process was one of the best project decisions I made. The starting GUI was created purely from code, while the game report GUI was created using Pygubu.

I finished it in a fraction of the time, with many additional features I would not have known about had I attempted to write it from scratch. The best aspect of Pygubu is the ability to see the adjustments made to a GUI in real time, without having to save, exit and run the code each time.

What was the most challenging aspect of the project?
Configuring and tuning the infrared signals was definitely the most difficult part of the project. I used the Linux Infrared Remote Control (LIRC) library which took the analogue and hardware side of transmitting signals and put it into software.

This was good because I wanted to keep the amount of physical components to a minimum. On the other hand, the library brought a great deal of struggles with it. There were many things the LIRC library provided, but a number of features I had to forego unless I added additional hardware to each Pi.

> "If I could turn on a fan using a dinky remote, then surely I could shoot information from a 3D-printed gun?"

Are there any things you'd change now, and are you planning to add any more features?
Because of the software limitations of the libraries I used, I would love to create an analogue version of the game. Instead of sending the infrared signals using the LIRC library, a more robust system could be made by adding a few components and sending the signals through hardware.

I also plan to implement additional game modes to add uniqueness and branch out from traditional laser-tag games. It was my intention from the beginning to have multiple game modes. All of the programming was done so that new game modes could easily be added to the existing code.

Make your own

Terran has included STL files on Instructables so you can 3D-print your own guns, along with a wiring diagram for the various components such as the LEDs, trigger and transmitter/receiver.

Keeping score

Terran installed Mosquitto MQTT Broker Service on an old desktop to act as a framework between the laser-tag game's server and the Pi-powered guns.

Components list

For three guns:

- Server (old desktop computer)
- 3 Raspberry Pi Zero Ws
- 3 4GB SD cards (minimum)
- 3 IR LED transmitters
- 6 IR Receivers
- 9 LEDs (3 red, 3 blue, 3 green)
- 3 passive buzzers
- 6 push buttons
- 3 LCD screens (16x2 character) with I2C adaptors
- 3 portable battery packs
- 3 micro to regular USB cables
- 3 PN2222 transistors
- 3 100Ω resistors
- 9 1kΩ resistors

One life left

The GUI on the server provides a lot more statistical information at the end of the game, but the gun's LCD display also supplies some useful information as you're playing.

Send the right signals

Terran ended up taking an IR LED transmitter from an old set of laser-tag guns. These had a black cone around the transmitter to help narrow the shot of each gun. If you want to replicate his work, Terran says that any general transmitter should do.

No shells required

Each gun has a Pi Zero W inside running Raspbian Lite, along with MQTT broker service for communication, I2C for the display libraries and Linux Infrared Remote Control.

1

2

Right between the eyes – or not

Accuracy results currently don't correlate to the actual receiver: "The original plan was to attach multiple receivers to each player," says Terran." This proved to be one of the limitations of the LIRC library that didn't function well with multiple IR receivers." Terran plans to build a hardware version to overcome this limitation.

Working out the kinks

Terran's top tip for anyone wanting to make a laser-tag game for themselves is: "Do your research first. Make sure you know the capabilities of MQTT, pygubu and LIRC before attempting to piece everything together. This makes debugging much easier when features are added and when things do not perform as expected."

Create a gaming controller prototype for the Pi

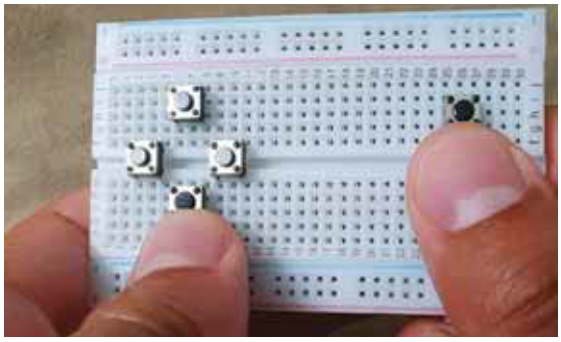
Using some cheap and readily available parts we're going to build a working video games controller

Calvin Robinson

Calvin works at an all-through school in North West London as the Associate Assistant Principal, Teaching & Learning, New Technologies.

Resources

- **Raspberry Pi**
- **Breadboard**
- **Jumper wires**
- **Tactile Push Buttons**
- **Pinout chart** https://pinout.xyz



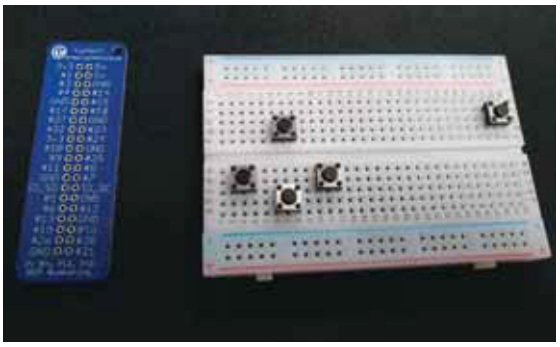
The Raspberry Pi is such a versatile little computer. Having access to the GPIO pins enables us to interface directly with any number of hardware peripherals, and in this case, create our own.

Using some basic bits of tech, including a breadboard, some jumper wires and tactile push buttons, we're going to build our own prototype video games controller that'll work with third-party Pi games, but could also come in handy if you're coding games of your own. Our prototype won't be pretty, but there's nothing to say you couldn't 3D-print a case and turn the controller into a legitimate-looking gadget.

The beauty of a project like this is that you don't need to be an electrical engineer; in fact, you don't need to know much more than the basics of an electronic circuit, because all the thinking is done with code in Python. So grab a few wires, some buttons and a breadboard, boot up the Pi and let's jump into Python IDLE.

01 Position buttons

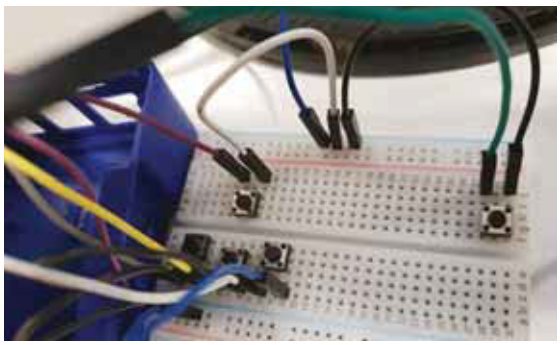
You'll need to decide how many buttons your controller is going to have. We're starting with five for simplicity: up, down, left, right and select – the latter being the default action button used for everything from starting the game to making a character jump. If you're building your controller to work with a particular game you may want to consider your options.



Place the buttons around the breadboard, leaving enough space to add two jumper wires per button: one for power and one for ground.

02 Connect the buttons

The next part can get a little messy. We need to connect the left side of each button to a GPIO pin on the Raspberry Pi (use the pinout chart to help with this). The GPIO pin will provide power and a method of control for our button using Python. The right side of each button needs to be connected to GROUND to complete the circuit. We could plug each cable into a different GROUND pin on the Pi, but in order to keep things tidy and minimise the number of wires between the Pi and the breadboard, we've linked multiple buttons to a row on the breadboard and then wired that to a GROUND pin on the Pi. We've done this once for the top and again for the button of the board, so we're only using two GROUND pins in total.



In the image you can see the black and white GROUND cables at the top of the breadboard are connected to the Pi via a blue wire. It'd make sense to colour-coordinate and do the same thing at the bottom.

03 Testing the connection

Now that all the hardware is connected we're going to want to test the connection of each button in Python. First, Import the `Button` module, then create a variable for each one, with an assigned GPIO pin number.

We've gone with BCM 2, 3, 4, 5 and 6 for the sake of simplicity for this project.

```
from gpiozero import Button
leftButton = Button(2)
```

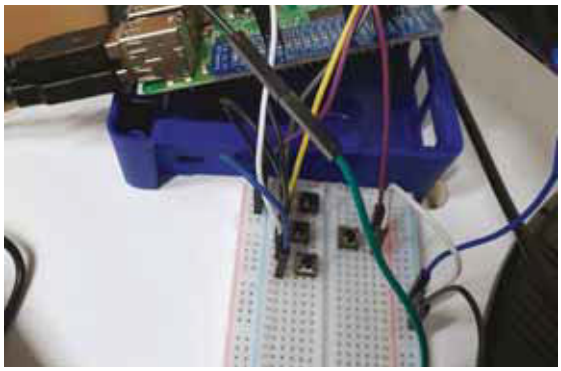
04 Add a loop and print command

Now that you've got the variables set up correctly, create a simple `while` loop and add each button with a simple `print` command to test the output. Press each button in sequence and see the text appear on the Python console.

```
while True:
    leftButton.wait_for_press()
    print("Left button pressed.")
```

05 Troubleshooting

The most common error we encountered while coding was something along the lines of: 'gpiozero.exc.GPIOPinInUse: pin 2 is already in use by <gpiozero.Button object on pin GPIO2, pull_up=True, is_active=False>'.



This means that we've accidentally used the same GPIO pin number more than once, and Python can't connect to it. Like all syntax errors it's an easy fix once you know where to look, double check the BCM numbers in the brackets of each `Button`.

It's also worth making sure everything is connected on the hardware side. Those jumper wires are only designed for creating breadboard prototypes, so they do tend to pop out on occasion.

06 A functioning controller, part 1

After testing every connection it's time to program our controller. We're going to want to create a function for the buttons:

```
def buttonFun(direction):
    if direction == "left": print("Left button pressed.")
elif direction == "down": print("Down button pressed.")
```

Add an `elif` for each button (down, right, select) to complete the function.

07 A functioning controller, part 2

Next, we'll need to add code to recognise the pressing of each button and pass a string to our newly created `buttonFun` function to let Python know which button we've pressed and print the appropriate text.

```
leftButton.when_pressed = buttonFun("left")
```

Likewise use the `when_pressed` method on `downButton`, `upButton`, `rightButton` and `okButton`.

08 Keyboard keys

At the moment our buttons function merely to print which button was pressed, but you might want to use this controller with third party games or software. We can do that by coding each button to represent a letter or function of a regular keyboard. We'd need to install `pyautogui` (`pip install pyautgui`) and import it:

```
from pyautogui import press, typewrite, hotkey
```

09 WASD

Now change each `print` command to `press` and assign the appropriate keyboard keys and we have a fully functioning game controller. Changing up, left, down, right to WASD and OK to SPACE creates a controller that would work in most games. Give it a go in Minecraft Pi Edition. Of course the real fun is creating your own game, so grab `pygame` (`pip install pygame`) and build something.

10 A fully functioning game controller!

With all this in mind we can add the following code to take care of the other button functions:

```
def buttonFun(direction):
    if direction == "left": press('a')
    elif direction == "down": press('s')
    elif direction == "up": press('w')
    elif direction == "right": press('d')
    elif direction == "select": press('space')
```

Once you've done this, your setup is complete!

GPIO label boards

While they're not a necessity by any means, we find it's always quite handy to have a GPIO label board to hand for these types of projects. Pop the board over your GPIO pins and you can instantly see which BCM is which, without needing to refer back to a print-out or website diagram.

The GPIO label board you'll see in our photographs is available for a whopping 38p online, so they're not exactly going to break the bank. One side is designed for the Raspberry Pi, Pi 2 and Pi 3, the other for Pi Zero models. Pop one on your keyring and you're always prepared for some Pi modding.

Dan Aldred

Raspberry Pi enthusiast, teacher and coder who enjoys creating new projects and hacks to inspire others to start learning. He's currently trying to get a Xbox 360 Kinect, Python and Pi to work with each other.

Resources

■ **Pimoroni Inky pHAT**
www.pimoroni.com

Getting started with the Pimoroni Inky pHAT

Install and try out a digital name badge, calendar project and more before starting on your own creation

The Pimoroni Inky pHAT is a beautiful 212x104-pixel, three-colour ePaper (eInk/EPD) display for your Raspberry Pi, and costs a very reasonable price as well at £22 ($25). These types of display are designed to emulate traditional ink on paper. Unlike conventional back-lit flat-panel displays that emit light, electronic paper displays reflect light.

In addition, the Inky pHAT maintains the image and text when the electricity is removed, thus acting as a static display. Best of all, because it's like paper, it's readable in bright sunlight which makes it suitable for outdoor use. It uses three colours – red, black and white – and is ideal for displaying simple graphics and crisply rendered text.

The Inky pHAT can be employed as a cute little clock, it can display tweets, updates from the current weather, news headlines, sports scores and so much more. It's also ideal for graphing data from remote sensors, CPU load or temperature, and stock prices.

In this tutorial we'll cover how to set up the hardware and install the required software. Then we'll move on to, and work through, a number of example programs such as a weather display, a calendar and a digital name badge. Once you're familiar with using the code and the display you can create your own image and display it on the screen.

Finally, we'll look at displaying text and writing this across several lines to fill the screen entirely. You can then take these skills and apply them to your own creations and projects.

01 Set up the hardware

As with all Pimoroni products, setting up your Inky pHAT is easy. Simply remove the product from the packing. Note that it ships with a pre-soldered header so it's ready to go, straight out of the box. Ensure that your Raspberry Pi is turned off, then place the Inky pHAT onto the GPIO pins, with the display in line with the USB ports. Gently push down to secure it into place. The pHAT is designed perfectly to fit the Pi Zero model, creating a slick and slimline hardware setup.

02 Install the software

Pimoroni has provided a software library to get you started with your Inky pHAT. It also includes a number of example programs. To install the libraries and

example programs open the LXTerminal window and enter the command line `curl https://get.pimoroni.com/inkyphat | bash` then press Return. Installation will begin and you will be asked to confirm the download and a few setting changes. Select the `Y` option and then press Return. If you'd prefer to manually install the software, for Python 3 use `sudo apt-get install python3-inkyphat` or for Python 2 `sudo apt-get install python-inkyphat`.

03 Make a digital name badge

Once the download has completed the Inky pHAT is ready to be used. The Pimoroni example programs are located in `/home/pi/Pimoroni/inkyphat`. They demonstrate a wide range of the capabilities of the hardware. In the LXTerminal navigate to the Pimoroni folder, type `cd /Pimoroni/inkyphat/examples` and press Return. The first program to try is the digital badge

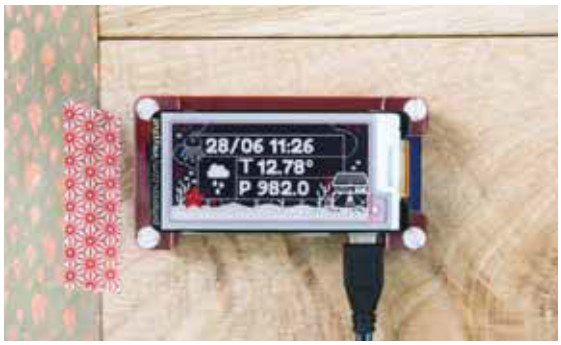
display. To run this type `python3 hello.py "your name"`, replace "your name" with a name of your choice, of course. Try your own to start with! Press Return and the Inky pHAT will begin to flicker as it loads the badge graphic and then adds your name to the display.

04 Display a calendar

One nice example program included in the examples folder is the graphical calendar. This program uses the code `now = datetime.datetime.now()` to read the current date and time from your OS and store the values in a variable named `now`. Some quick maths later and the values are combined with the width and height of each segment of the month, which represents the days, before highlighting with a white square the current day and date. To run this example, type `python3 calendar.py` and press Return.

05 Display the weather

The Inky pHAT is the perfect size to make a dinky weather display and guess what, the examples folder includes one. Open the LXTerminal window and type `sudo pip install requests`. This enables Python to access and pull down data from a weather website. Next, open the `weather.py` program in your Python editor and scroll down to line 26. Here you will find the location



details and settings for the weather. Change `city =` to your current city. Then add your country code, which is currently set to Great Britain (GB). Finally, adjust the maximum temperature warning. This will currently be set at 25, but adjust the value to reflect your environment. Save the file and return to the terminal window, type `python3 weather.py` to load and run the graphical weather display.

06 Display the Inky pHAT logo

The `logo.py` program takes a PNG image file and writes it to the display. In the example program the image is a file named `InkyPhat-212x104.png`, which is the logo for the hardware product. The image measures 212x104 pixels and displays the full use of red, black and white colours. In the LXTerminal window, type `python3 logo.py` and press Return. The Inky pHAT will begin to flash and then the back elements of the image will be written to the display followed by the red elements. In **step 8** we'll look at how to create and display your own image.

How E-ink works

E-Ink actually uses real ink to display text and images. The display consists of a capsule film which holds a clear fluid containing tiny particles. Electrodes are used to apply a positive or negative electric field, and colour particles with the corresponding charge move the top of the capsule, making the surface of the 'paper' appear a particular colour.

07 Cleaning the screen

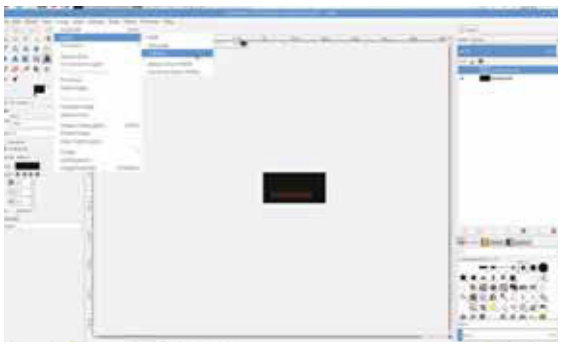
Over time the Inky pHAT will encounter screen burn – a discolouration of areas on the e-paper display. This occurs over time due to non-uniform use of the pixels. Fortunately, this can be resolved using the cleaner program in the **examples** folder. This basically displays solid blocks of red, black and white to 'clean' the Inky pHAT's display. In the LXTerminal window type `sudo clean.py` and press Return. The program takes a little while to complete, but do leave it to finish.

08 Create your own image: part 1

You can use most image editing programs to create your image. However, if you want to create the image on your Raspberry Pi then install and use GIMP. In the LXTerminal type `sudo apt-get install gimp` and once it has installed type `gimp` to open it. Once it loads select File > New, set the size of the canvas to width 212 pixels and the height to 104 pixels. You can now begin to create your image, remembering that you only have black, white and red available.

09 Create your own image: part 2

You can add text to your image by selecting the large back capital A from the left toolbar. Click the canvas and a text box will appear. Type in your text, editing the font and size as required. Again you can change the colour to either black, red or white. Once you have completed your image you need to convert the colour palette to the three colours. Open the LXTerminal and download the Inky pHAT color palette; type `git clone https://github.com/pimoroni/inky-phat` to clone the GitHub repo. You'll find the colour palette saved in the folder as `inky-phat/tools/Inky-pHAT.gpl`.



10 Create your own image: part 3

In GIMP, return to the Image menu, then Mode > Indexed. Select the option for a 'custom palette' and click the Palette selection dialogue button at the bottom-right of the window. When the palette window loads, right-click and select the Import Palette option. Now navigate and select the `Inky-pHAT.gpl` palette file from the download in **Step 9**. You can export the image as a PNG file. Click 'File', scroll down the menu and select the 'Export as' option. Save and name your file into your folder. A dialogue box will appear with a number of saving options; ensure that you tick 'Save background color', then click 'Export'.

11 Display your own image

To display your image on the inky pHAT open your Python3 editor and write a short program. Begin by importing the `inkyphat` module and then the `PIL` library. Then use `inkyphat.set_image()` to load your image, replacing the `inky.png` filename with the name of your file. Save the program and then execute the program. Your image will be displayed on your inky pHAT.

12 Displaying text: part 1

One of the common uses of the Inky pHAT is to display messages or updates and this can be achieved with text. However, you may not always know what the length of the text string will be – it could be 15 characters or it could be 100 or more. Whatever the length you need a program to display the text accurately in the centre of the screen. Open a new Python 3 file and import the following modules. `ImageFont` enables you to select a font style of your choice, and `textwrap` enables you to wrap the text over several lines.

```
import sys
from PIL import ImageFont
import inkyphat
import textwrap
```

13 Displaying text: part 2

Now we'll set the font style by creating a variable called `font` and applying the Fredoka font. The number 14 refers to the size of the font; this can be adjusted to fit your requirements and your eyesight. On the next line we create a variable to hold the message you want to display on the screen. With the current settings you will probably

be able to display around 250 characters before the text becomes unreadable. The last line sets the spacing and is relative to the font size. A value of 30 or 31 renders the text with accuracy; if you change the font size remember to adjust this value too.

```
font = ImageFont.truetype(inkyphat.fonts.FredokaOne, 14)
message = "This is an example of text"
txt = textwrap.fill(message, 31)
```

14 Displaying text: part 3

When the text is plotted onto the screen it uses an x and y co-ordinate position. A single word will be displayed in the centre of the screen; a longer string of text needs to be broken down into lines. The following code takes the text string and breaks it down into lines based on the required height and width of the text that you set in the previous step. Most of the time this renders all the text readable and presented within the inky Phat screen. However, if you change the font size remember to adjust the `textwrap.fill()` value too.

```
w, h = inkyphat._draw.multiline_textsize(txt, font)
x = (inkyphat.WIDTH / 2) - (w / 2)
y = (inkyphat.HEIGHT / 2) - (h / 2)
```

15 Displaying text: part 4

These last two lines of code write the text to the screen using the x and y values calculated in the previous step. The ink colour is set to black and the font type from **Step 13** is applied. The final line of code shows or writes the text to the screen. Remember that if the text is not properly displayed you can adjust the font size and the `textwrap.fill()` value until it looks correct. You can also change the ink color to `RED`.

```
inkyphat._draw.multiline_text((x, y), txt, inkyphat.BLACK, font)
inkyphat.show()
```

16 Run the program

Once you have completed the Text display program, add a suitable line of text to line six of the program, which is then stored in the `message` variable. Press F5 to save and run the program. Your message will be written to the Inky pHAT. You can change the content of the message as you like; just re-run the code to update the display. ■

Uses for the inky pHAT

The Inky pHAT is ideal for displaying information or data in real time. For example, why not create a Twitter display that notifies you of your incoming tweets, mentions and hashtags? If you're an avid surfer, sailor or just generally like the sea then why not build a tide display. This project pulls down tide data from the Met Office and presents it in a good-looking way: **http://bit.ly/lud_tides**. Finally, if you like your art then consider making some graphics of your favourite game, film or anime character: see **http://bit.ly/lud_art** for a tutorial. You could probably even create digital Inky pHAT versions of some classic paintings such as the Mona Lisa, The Starry Night or The Scream.

MAKE YOUR OWN PROJECTS WITH THE RASPBERRY PI

Learn the electronics, computing and coding skills you need to make your own projects with the Raspberry Pi, and let your imagination run wild

FUTURE

Ordering is easy. Go online at:

www.myfavouritemagazines.co.uk

Or get it from selected supermarkets & newsagents

Reviews

81 **Group test** | 86 **Hardware** | 88 **Distro** | 90 **Free software**

GROUP TEST

Recovery distributions

Linux is very resilient, but when things go wrong you can use one of these specialised distributions to repair and rescue a broken installation

ALT Linux Rescue

The ALT Linux project produces a bunch of distributions for different purposes, including fully stocked desktop and server distributions. It also has a specialised version named Rescue that works on 32-and 64-bit computers and helps system admins fix their computers.
https://en.altlinux.org/Rescue

MorpheusArch Linux 2.0

MorpheusArch Linux is built on Arch Linux. The Live distribution is relatively new compared to its peers and is only available for 64-bit machines. It uses the lightweight and elegant LXQt environment and has several tools to help admins.
www.morpheusarch.co.uk

SystemRescueCd 5.2.2

One of the oldest distributions that's designed to equip system admins with a variety of tools and utilities to help repair and rescue data from Linux or Windows. While it's meant to be used as a Live CD you can also anchor the distro to your hard disk.
www.system-rescue-cd.org

Ultimate Boot CD v5.3.8

Another Live distribution that's packed full of backup and recovery tools, Ultimate Boot CD (UBCD) began because its author wanted to run a floppy-only diagnostic tool from an optical drive. That initial success led UBCD to grow even further.
www.Ultimate Boot CD.com

ALT Linux Rescue

Loaded with tools, it's ideal for experienced sysadmins

Inserted module 'scsi_dh_alua'
Inserted module 'scsi_dh_emc'
Inserted module 'scsi_dh_rdac'
Inserted module 'dm_multipath'
Updating chrooted environments: [DONE]
INIT: Entering runlevel: 3
Entering non-interactive startup
Starting network: [DONE]
Starting rpcbind service: [DONE]
Initializing random number generator [DONE]
Starting system logger service: [DONE]
Starting kernel logger service: [DONE]
Handling remaining udev events: [DONE]
Configuring network: eth0 [DONE]
ssh-keygen: generating new host keys: RSA1 RSA DSA ECDSA ED25519
Starting sshd service: [DONE]
** WARNING: remote root access is now enabled
** IP(s): 192.168.0.7
** sshd port: 22
** root password: tree@Ea1m
More info: http://en.altlinux.org/rescue

ALT Linux Rescue has a boot option that'll bring up networking and start the SSH server for remote access to the machine

Tools repository

ALT Linux Rescue packs in over a hundred CLI tools, including several versatile ones for stress-testing hardware and benchmarking a system. The distribution also has several utilities to help recover deleted files and fix common issues with both Linux and Windows installations. Besides these there are several others available in its official repository.

Custom tools and UIs

The distribution doesn't offer much in terms of customisation. One of the few aids offered is the forensic-mode boot option that brings up the distribution without activating the swap or mounting any partition. There are also a couple of scripts to automatically mount attached disks and fix any issues with the master boot record.

User experience

ALT Linux Rescue boots straight to the shell. Besides the couple of helpful scripts mentioned earlier, there are no other conveniences. The distribution is designed for someone familiar with the bundled tools and who knows the ones they need to use to fix the issues affecting their system. Unlike some of its peers, it's missing a graphical desktop.

Support and rescue docs

The distribution only gets a single page on the ALT Linux website, which lists the Live CD's capabilities and also includes a software spotlight section that lists some of the marquee utilities. There are Russian-forums and mailing lists for the main ALT Linux distribution but the Rescue edition doesn't have any official support avenues.

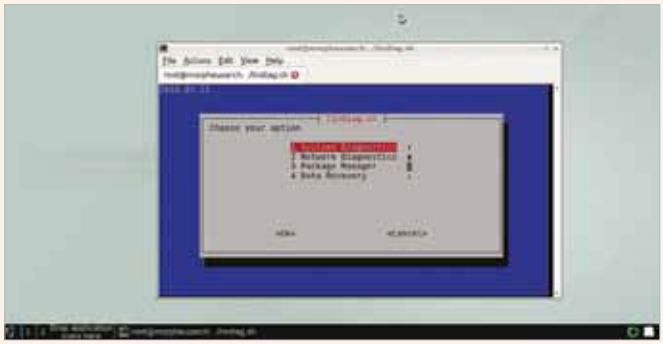
Overall

ALT Linux's Rescue edition is designed for experienced sysadmins who need an all-in-one Live environment to fix issues. You won't get much mileage if you aren't familiar with the tools.

4

MorpheusArch Linux

A distro that's good for recovering data, but not much else



Besides MorpheusArch, you can use the LinDiag utility on a regular Arch installation as well as on Fedora and Debian

Tools repository

The distribution has a very focused approach and only includes testdisk and photorec utilities for recovering files. The tools support all the popular filesystems and they can identify files in over 400 formats. In addition to files, you can also use them to find lost partitions, rebuild the boot sector, fix FAT tables, locate the backup superblock and more.

Custom tools and UIs

The MorpheusArch project produces an ncurses-based script called LinDiag to assist with common system admin and recovery tasks. You can use it to check services, diagnose the network, manage packages and recover data from drives – but it isn't included in the distro by default and you'll have to clone it from GitHub.

User experience

Compared to some of the other distributions, the appeal of the distribution is fairly limited. It boots up to a minimal LXQt desktop with a smattering of apps, including the text-based elinks web browser. There's no indication on how to proceed and you are also supposed to manually pull-in the assistive LinDiag utility, which is also not very apparent to new users.

Support and rescue docs

One of the two distributions in this group test that doesn't have any sort of help information of its own. You do get a brief illustrated guide on LinDiag's GitHub page, but there are no forum boards or mailing lists on which to seek help, so this distribution is only designed for people who want to use the bundled recovery tools on an Arch base.

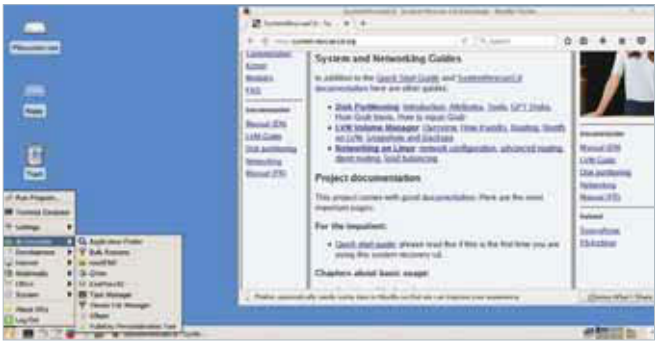
Overall

MorpheusArch Linux is designed for Arch users who are familiar with the CLI-based data-recovery tools that it bundles. Its assistive LinDiag script isn't very useful unless it's bundled along with the distribution.

5

SystemRescueCd

A good option if you can use the tools or have the patience to read the docs



SystemRescueCd ships with a minimal Xfce desktop, along with some essential apps such as Firefox, KeePassXC and others

Tools repository

The distribution packs in several repair and rescue tools and utilities, and supports all the important filesystems. Besides CLI utilities, it includes disc images for several tools such as FreeDOS, NTpasswd, MHDD and so on. There's also FSArchiver, which can help save and restore a partition table, or the contents of a file system, to a compressed archive file.

Custom tools and UIs

SystemRescueCd supports loading SRMs (System Rescue Modules) that are useful for adding new applications to the distribution. In addition to apps, SRMs can be used to add custom data files. Further customisation is possible with the sysresccd-backstore utility, which helps build a loopback file system to which you can save custom files.

User experience

The distribution has a comprehensive boot menu. Besides the dozen options listed on the main screen, there's help information explaining the various advanced boot parameters over several virtual consoles. The default boot option boots into a console which lists some basic commands to configure the network, mount NTFS partitions and start the graphical environment.

Support and rescue docs

The project hosts a quick-start guide as well as detailed instructions on basic and advanced use. There are also instructions for experienced campaigners, such as the guide on how to make a custom version and backing up data from an unbootable Windows computer. The boot menu also offers troubleshooting suggestions.

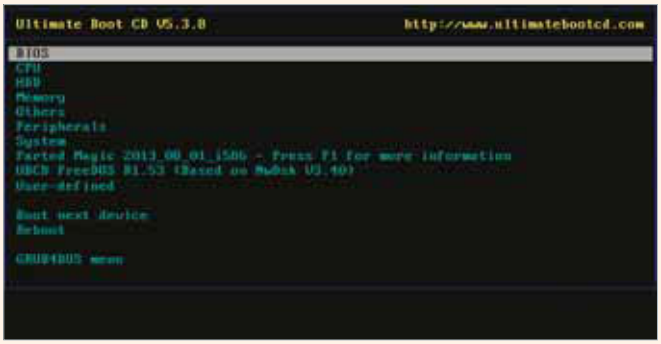
Overall

A well-rounded recovery solution that'll help resolve most common issues. It has enough docs to guide inexperienced users, while advanced ones will appreciate its customisation options.

8

Ultimate Boot CD

A CLI-only system that's surprisingly easy to navigate and use



UBCD's menu divides the packaged utilities into categories based on the system area they influence, such as BIOS, HDD and memory

Tools repository

The distribution is loaded with tools. You can use UBCD to query and stress-test hardware and peripherals connected to a system. It can be used to restore BIOS settings as well as restore and backup the CMOS. UBCD enables you to fix bootloaders, recover lost passwords and deleted files. It will also help you tweak the Windows Registry without booting into the installation.

Custom tools and UIs

UBCD's biggest customisation option is its custom boot menu. The distro ships with no graphical desktop and instead boots to a very well laid-out menu. The customised text-based menus help guide users to the relevant utility for their issue. The menu is logically arranged and individual entries have useful information to help identify them.

User experience

As we mentioned, UBCD boots to its customised text-based menu, which divides the packaged utilities into categories based on the part of the computer that they influence. The menu might be text-based, but the clear categories and helpful descriptions mean that the distro can be easily navigated and used by inexperienced users to fix all kinds of issues.

Support and rescue docs

UBCD's website hosts a fairly detailed FAQ to familiarise users with the distribution. However, the tutorial section is really just a pointer to a long list of user-contributed tutorials curated from the forum boards. The documentation on the project's wiki isn't of much help either, as most of it is incomplete and isn't intended for new users.

Overall

Lives up to its name with its ultimate collection of tools and utilities to troubleshoot problems in every component of the system. The lack of a graphical desktop is no impedence.

8

In brief: compare and contrast our verdicts

	ALT Linux Rescue	Morpheus Linux	SystemRescueCd	Ultimate Boot CD
Tools repository	Includes more than 100 CLI tools and a couple of scripts to automate tasks **8**	Has a very focused approach and includes a limited number of tools, for recovering data **6**	Packs in loads of useful repair and rescue tools, along with several disc images **8**	Lives up to its name and is overflowing with all kinds of system tools and utils **9**
Custom tools and UIs	Doesn't ship with a graphical desktop, and bundles only a few custom scripts **5**	The distro can work with the LinDiag script, but this isn't included in the Live CD by default **7**	There's support for modules which are useful for customising entire distributions **8**	Uses an extensively modified boot menu that helps select the right tool **9**
User experience	Boots straight into a shell and expects its users to know their way around it **5**	It ships with the LXQt desktop but includes virtually no UI-based apps to use this **6**	Has a comprehensive boot menu and the shell also displays useful info on boot **8**	UBCD's customised boot menu makes it fairly simple to narrow down tools **9**
Support and rescue docs	There's hardly any documentation and no avenues for seeking support **0**	Offers no documentation besides the minimal usage information for LinDiag **4**	Ample documentation for users new to the project, as well as experienced ones **8**	Has enough documentation and a support forum to help orientate new users **7**
Overall	**Meant for experienced users who are familiar with the bundled tools** **4**	**An Arch-based distribution with a nascent but still useful graphical tool** **5**	**Together with its docs, it's a nice package to help you wriggle out of many common issues** **8**	**True to its name, UBCD offers a lot of tools via a user-friendly interface** **8**

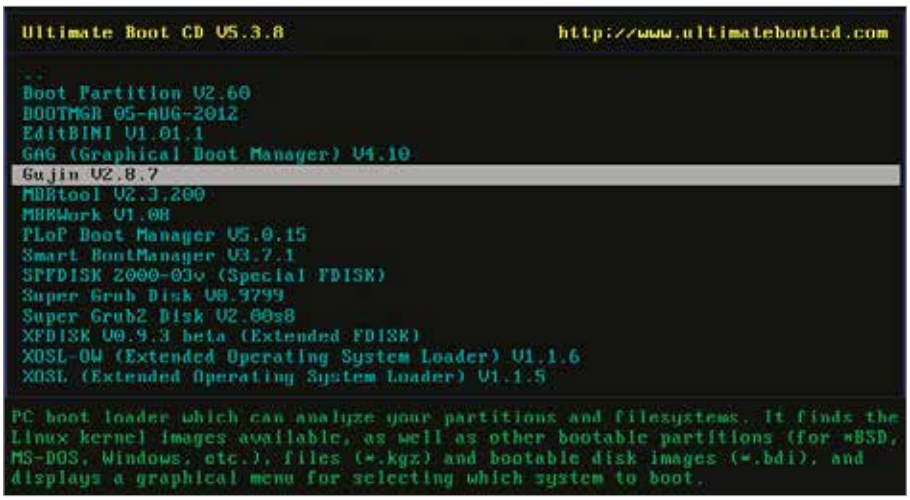
AND THE WINNER IS...

Ultimate Boot CD

To help cure your system of common ills, you need a distribution that has the right tools. But just having the largest collection of tools isn't of much use in the hands of an inexperienced user. This is why one of our top objectives is to find a Live CD that can be used by the largest number of people, irrespective of admin skills and experience.

It's because of this that ALT Linux loses out. Although it has a wide range of apps, in the absence of a graphical environment or customised wizards it can only really be used by experienced campaigners who know which tool they need to use and are familiar with its usage. MorpheusArch Linux loses out for similar reasons. Though it boots into a graphical desktop, the distribution doesn't have any interface to the CLI tools. The project's LinDiag utility is a nice attempt to make some of the tools more palatable to first-time users, but it only offers a very limited number of options and isn't included in the distribution by default.

It's a close contest between SystemRescueCd and UBCD, both of which use customised boot interfaces.


Ultimate Boot CD V5.3.8
http://www.ultimatebootcd.com
Boot Partition V2.60
BOOTMGR 05-AUG-2012
EditBINI V1.01.1
GAG (Graphical Boot Manager) V4.10
Gujin V2.8.7
MBRtool V2.3.200
MBRWork V1.08
PLoP Boot Manager V5.0.15
Smart BootManager V3.7.1
SFFDISK 2000-03v (Special FDISK)
Super Grub Disk V0.9799
Super Grub2 Disk V2.00s8
XFDISK V0.9.3 beta (Extended FDISK)
XOSL-OW (Extended Operating System Loader) V1.1.6
XOSL (Extended Operating System Loader) V1.1.5
PC boot loader which can analyze your partitions and filesystems. It finds the
Linux kernel images available, as well as other bootable partitions (for *BSD,
MS-DOS, Windows, etc.), files (*.kgz) and bootable disk images (*.bdi), and
displays a graphical menu for selecting which system to boot.

UBCD includes tools to stress-test hardware as well as a host of other useful utilities

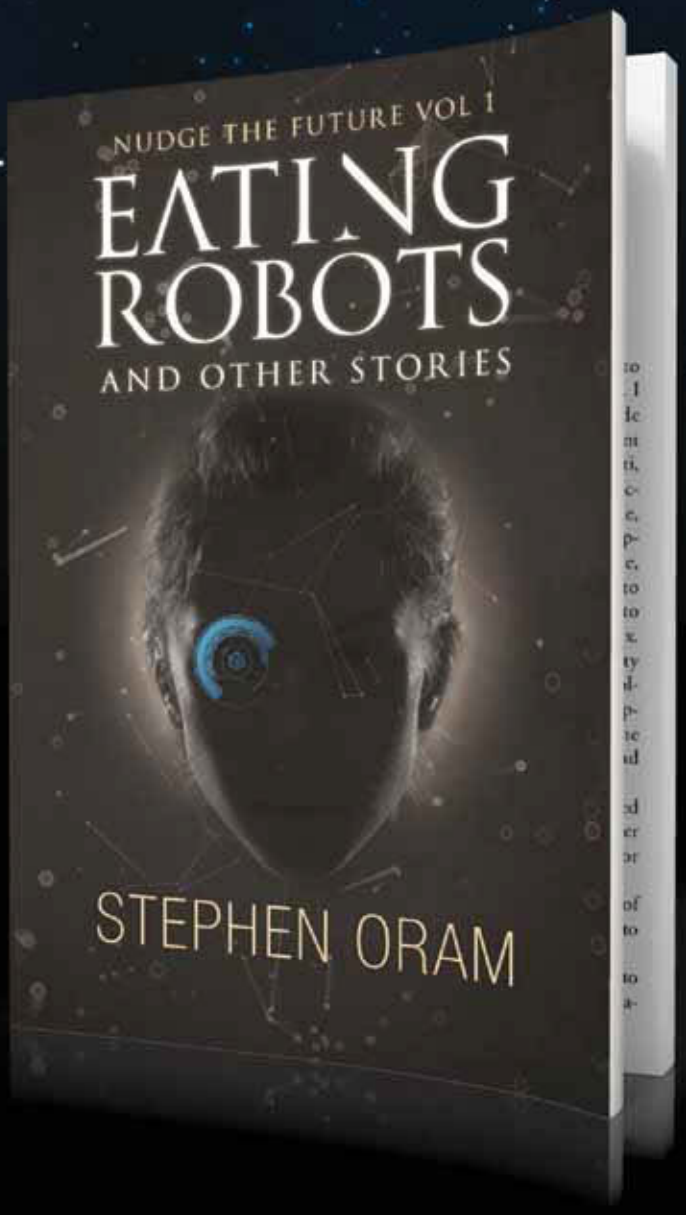
SystemRescueCd lists a number of boot options and also offers useful help information over different virtual consoles. Still, using SystemRescueCd isn't as intuitive as UBCD, whose boot menu is logically arranged and each option has useful information to help users narrow down to the right tool and then boot straight into it.

Please be careful when using the repair and recovery tools in these distributions. It'll be prudent to spend some time reading through their documentation before attempting to use them. If used improperly, they can in fact cause more damage and complicate the recovery process.

Mayank Sharma

EATING ROBOTS
AND OTHER STORIES

The future is ours and it's up for grabs
Robot-driven redundancies, extreme health monitoring, rogue algorithms, fake news, hacked driverless cars...



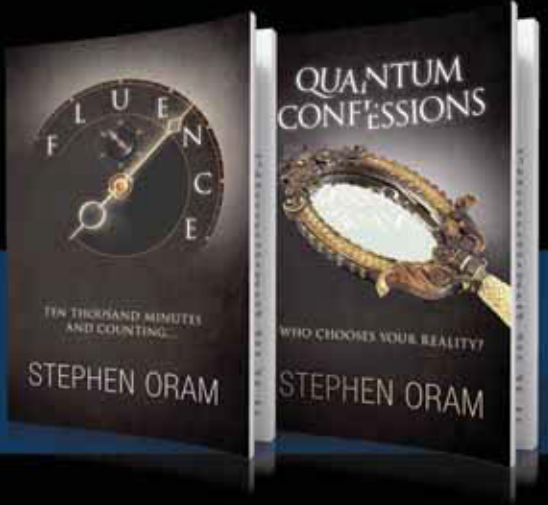
"If you are interested in technology, shows such as Black Mirror or just in understanding the future, I can't recommend Eating Robots and Other Stories enough. It is a fantastic read from start to finish!" – Gamified.co.uk

"An insightful, often worrying, set of thought experiments on the possible unintended consequences of near future AI." — Alan Winfield, Professor of Robot Ethics at UWE, Bristol

"A fizzingly inventive collection of nearly three dozen short stories from an author rapidly establishing himself as the leading voice on how technology may determine the ways in which societies and individuals are structured in the years to come." – Morning Star

"A well-crafted, highly imaginative collection of ideas that are inescapably relevant to our era. At times it is terrifying. Other times it is whimsical. But every instalment is food for thought."
– Orchid's Lantern

Available at Amazon and all good bookshops in paperback and ebook



Also by Stephen Oram: Fluence and Quantum Confessions

Above If you're looking for a tough notebook without shelling out for a 'premium' version, the Acer could be ideal

HARDWARE

Acer Chromebook Spin 11

Acer spins out a winning ChromeOS machine

Price
£329 ($349)
Website
www.acer.com
Specs
CPU 1.1GHz Intel Celeron N3350
Display 11.6-inch 1,366 x 768 IPS touch
Graphics Intel HD Graphics 500
RAM 4GB LPDDR4
Storage 32GB eMMC SSD
Ports 2x USB Type-C, 2x USB 3.0, microSD, audio jack
Connectivity 802.11ac wireless LAN, Bluetooth 4.2

In the past, Chromebooks have generally been regarded as lower-end cheap laptops, fit for education and casual web-browsing. However, in a post-Pixelbook world, do they still fill the same role? Well, as the Acer Chromebook Spin 11 proves, they certainly can.

It's designed to appeal to parents with young children that need a laptop to do schoolwork, but it's also suitable for someone who wants to muck about installing Linux. It's durable, has decent enough performance, comes with an included stylus and carrying case – and it's inexpensive considering all that's in the box.

There's only one configuration available for purchase. Just £329 ($349) nets you the laptop with the specs we list, which are fairly average for the category compared to similar Chromebooks – such as the Dell Chromebook 11 3180 at £240 ($319), with a less powerful CPU and the same amount of memory and storage.

The Spin 11 features an all-plastic build, but don't let that deceive you as this Chromebook is one of the sturdiest laptops we've ever used. No matter how hard we tried, we couldn't get the laptop to flex even a little. On paper, the keyboard would also seem shallow with just 1.1mm of key travel, but it feels comfortable and we didn't feel the keys bottoming out at all while working on it.

It's not the lightest Chromebook in the shop, weighing in at 1.35kg. However, its thin 2cm frame with rounded corners makes it pleasant enough to hold, and we're sure that it was difficult to make it

Above The inclusion of two USB-C ports for data transfer and charging is a welcome touch

> It's durable, has decent enough performance, comes with an included stylus and carrying case

much lighter without giving up some of the durability. Ports are also a huge win here – you have two USB-C ports, which handle data transfer and charging. In a Chromebook at this price range, we would have been happy with just one. You've also got two USB 3.0 ports, a microSD card slot and a headphone jack.

The screen also reflects (if you'll excuse the non-pun) the durable nature of the Chromebook, as it's covered by Corning Gorilla Glass, so it should be pretty resistant to cracking or shattering. The touchpad is a sour point, however. The surface is competent, but it's extremely sensitive, not to mention that the plastic finish and tough centre-click was a turn-off. However, it does support gestures, so it's not all bad.

Things start to take a turn when you look at the display. It features a 1,366x768 IPS display, and while viewing angles are quite good, it's not very bright. At this price and form factor an HD display isn't exactly out of the ordinary, but we would have appreciated a brighter panel, if not Full HD resolution.

The speakers, also, are a bit lacklustre; while they do manage to fill the room, there's not all that much in the way of detail in the sound. So the Spin 11 is fine for watching some YouTube videos; just don't expect a pleasant experience while listening to music or watching 'real' films. The Acer isn't a powerhouse, and we weren't expecting it to be.

Still, even with its modest components, Chrome OS continues to prove itself competent for its intended use cases: generally web browsing and word processing.

While it won't be able to keep up with something like the Google Pixelbook, rocking an Intel Celeron processor as it does, it was able to keep up with a decent workload: typing this review with 10 tabs open in Chrome and a music player running in the background. Quite honestly, that's exactly what we wanted to see from this device.

However, one of the most important aspects of a Chromebook is its battery life, and unfortunately the Spin 11 disappointed here. While in our casual use we didn't really need to stress about charging the device, it didn't exactly perform admirably when we looped a local video at 1080p running in VLC: the device died after just 7 hours and 34 minutes.

For a Chromebook, this score is well below the category average. For instance, the similarly configured (and older) Dell Chromebook 13 scored almost double when running the same test, at around 14 hours.

Overall, then, the Acer generally falls in line with similar devices in the US. But when you take the included stylus and carrying case into consideration, it actually starts to look like a bargain.

Bill Thomas

Pros

An affordable price for a Chromebook which comes with a stylus and carry case. It also boasts a durable frame design that'll last for years.

Cons

The IPS screen isn't as bright as we'd like (and you can forget about Full HD resolution), its battery life is less than stellar, and performance is merely adequate if actually a little lacklustre.

Summary

If you're looking for a cheap laptop for education or for general browsing, and you don't need frivolous features and a high-resolution display, the Acer is a great device. Just bear in mind the lower-than-average battery life.

8

Above If you have an Nvidia graphics card, the distribution also features Driver Manager, which can help you install the driver for your specific card

DISTRO

Linux Mint 19 Cinnamon 'Tara'

Specs
CPU Any AMD, VIA or Intel processor
Graphics 1024x768 resolution
RAM 1GB RAM
Storage 15GB disk space for minimal install
Licence GPL
Available from
https://linuxmint.com

The latest LTS release has had long gestation period and it's yet another solid release

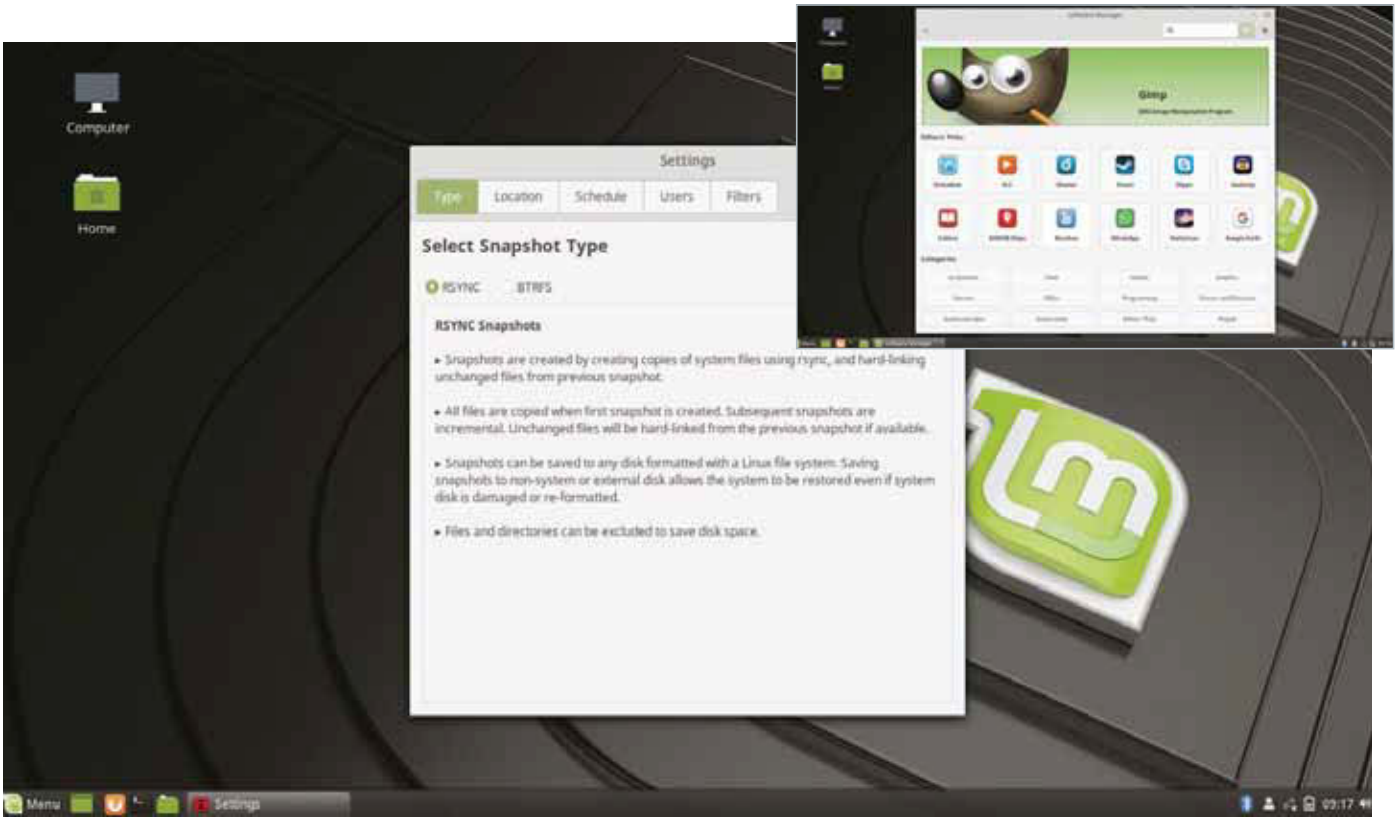
Originally based on Debian, but now built on Ubuntu's LTS releases, Mint follows a 'release when ready' cycle, instead of a fixed schedule as preferred by Fedora, Ubuntu and many others.

Codenamed Tara, the latest release of Linux Mint ships after nearly a month spent fixing the myriad bugs reported in the beta release. In a blog post, the Mint Team expressed gratitude to the community for the steady stream of bug reports on the beta, which helped the developers take corrective measures. This nearly zealot-like community involvement and the tireless efforts by the developers are the reason for Mint's soaring popularity, and has helped position the distribution as one of the most newbie-friendly distros available.

With Mint 19, the distribution has dropped its KDE edition, and now produces three variants favouring Cinnamon, Mate and Xfce desktop environments. The project still provides installable live images for both 32- and 64-bit machines.

The homegrown installer enables you to add proprietary codecs and drivers for devices such as graphics cards, but users can also use the 'Install Multimedia Codecs' utility to add these post-installation. You can choose to encrypt the root partition, which requires entering a passphrase, or alternatively only encrypt the home directory. The release notes, however, advise against the latter option, as it's slower than whole-disk encryption.

Mint 19 boots to the desktop in just a little over 10 seconds on a machine with just 4GB RAM. Unfortunately, the performance is only slightly better with 16GB RAM. While the boot time is on par with similar desktop distros such as Ubuntu, OpenSUSE and Fedora, it's slower than certain rolling-release variants such as Manjaro Linux and Solus.



Above Taking snapshots of your installation is easy with the built-in Timeshift tool

> If you want to spruce up the look of your installation, the Extensions utility enables you to add a large number of effects such as wobbly windows and workspace scroller

In addition to a custom beginner-friendly backup utility, the latest release of Linux Mint also features Timeshift. The tool, which was originally introduced to the Mint community with Mint 18.3, is designed to help you create snapshots of your working installation, using either rsync or btrfs. Should you accidentally break the system, such as when you're testing experimental software, you can revert the system to a previous snapshot state. Best of all, you can even automate snapshots on a weekly, monthly, or even hourly basis depending on your needs. You can also choose to make only incremental snapshots with rsync, which helps save disk space.

Several essential components such as the Nemo file manager and the Update Manager have undergone improvements. The latter now advises users to deploy all updates as they become available, since system stability and longevity is guaranteed by Timeshift snapshots. Should your installation grow to include packages from different third-party repositories and PPAs, the Update Manager now identifies the source of each of the available updates.

If you want to spruce up the look of your installation, the Extensions utility enables you to add a large number of effects such as wobbly windows, a workspace scroller, a Compiz-like 3D cube and so on to the desktop.

While the distribution still includes Synaptic, Software Manager is the primary application for handling software. Unlike its peers on other distributions, Mint's Software Manager can easily be navigated using just the keyboard, and now supports searching within categories.

With its focus on performance, and the inclusion of a backup and snapshot tool out of the box, Mint 19 is a polished release and should appeal to novices and experienced users alike. Although APT remains the underlying package-management system, the distribution continues to improve its support for Flatpak with the latest release, now featuring single-click install of packages.

Thanks to a long history of stable releases under its belt, Mint has created a distinct identity for itself, which is rare for derivative distributions.

Shashank Sharma

Pros

A strong user community, dedicated devs and plenty of custom apps make this ideal for everyone.

Cons

The distribution is missing several useful applications – such as an IM client – out of the box.

Summary

Linux Mint has a history of producing well-rounded editions. The latest LTS release will be supported until 2023 and features a number of custom tools such as the Software Manager, and useful new additions like the Timeshift snapshot tool for troubleshooting.

9

DESKTOP SHELL

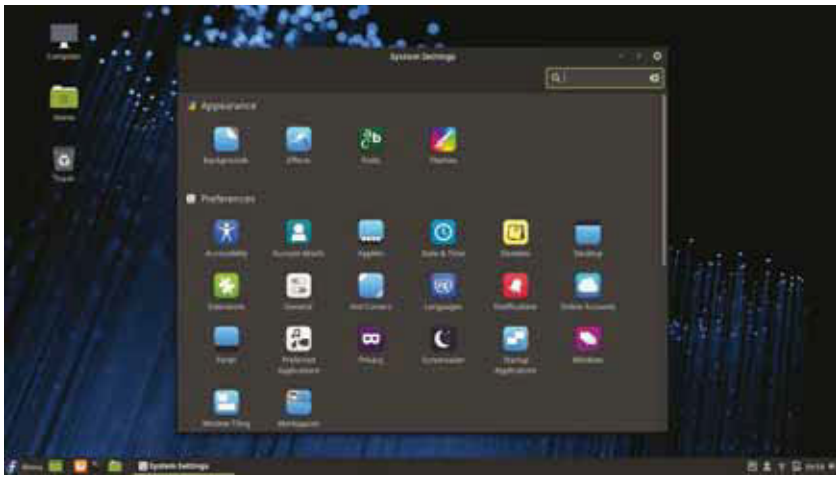
Cinnamon 3.8.4

The visual component of the desktop gets an update

Cinnamon is the shell or user interface of the popular Cinnamon desktop. This component is responsible for the various elements you see in the desktop environment, including panels, menus, hot corners and such. Cinnamon's user interface is written in JavaScript and its core libraries are written in C.

The Cinnamon project is headed by Linux Mint, but the desktop is developed separately from the distribution. One of the main reasons for the desktop's popularity is that it adheres to the traditional desktop metaphor with a single panel located at the bottom of the screen, a design reminiscent of GNOME 2. The desktop environment is made of several components such as the Muffin window manager, the Cinnamon Control Center and the Cinnamon shell.

Most of the work in this minor new release is behind the scenes, though the effect of some of the changes are visible on the desktop. For example, the network applet no longer displays unmanaged networks, and system settings now sports a keyboard backlight section. The update should have made its way into the repositories of your distribution, so you should switch to it as soon as you get the notification.



Above Don't confuse the Cinnamon Shell with its namesake, the Cinnamon Desktop

Pros

Offers a traditional desktop layout without skipping various modern functionalities.

Cons

Although it's not as demanding as other desktops, you can't run it on low-spec machines.

Great for...

The best of both worlds: traditional layout and modern features.

http://bit.ly/lud_cinnamon

PROGRAMMING LANGUAGE

Crystal 0.25

Aims to combine C's speed with Ruby's slickness

Crystal is a fairly new language which was developed to offer a Ruby-inspired syntax with a fast compiler that produces efficient native code. It seems to working too, because according to benchmarks by one programmer (**https://github.com/kostya/benchmarks**) Crystal is an order of magnitude faster than Ruby.

Unlike Ruby, though, Crystal is a compiled language. Programmers will appreciate this because it helps them catch errors at compile time instead of causing issues when the code is put into production.

This also means that Crystal enables you to ship binaries – so if the target has limited storage space, you can avoid copying the whole repository and just use the pre-compiled binary instead.

Programmers will also appreciate the fact that Crystal is statically typed. Most languages don't do any kind of type checking, but with Crystal you can add type definitions to your code and you get the flexibility to use a combination of types. That said, specifying types isn't strictly necessary because the language uses an advanced algorithm to infer types automatically from context.

Crystal is released under the Apache Licence v2.0. Head to the installation section on its website to add its repository for your distribution. You can then pull the latest version of Crystal's compiler using your distribution's package manager.

The documentation on its website will help you get started with using Crystal and also has details about its syntax and semantics.

Pros

Does a nice job of combining the elegance of a Ruby with the efficiency and speed of C.

Cons

Not everybody likes a compiled language and being Linux-only limits its uptake somewhat.

Great for...

Writing fast and efficient code with the fluidity of Ruby, if you don't mind the extra step of going through compilation.

https://crystal-lang.org

FTP SERVER

uftpd 2.5

A 'no-nonsense' FTP server which is just that

Linux has no end of good FTP servers, but uftpd has carved a niche for itself by shipping with sensible defaults which work for most users. It's designed for home users and developers who need a simple FTP server, but aren't too particular about being secure. In fact the documentation of the project advises against using the server if security is a deciding factor. On the other hand, uftpd isn't *totally* insecure and its author has implemented some common security best-practices (though not all of them, in the interests of keeping the program more user-friendly).

Combined with users taking adequate precautions and defining strict permissions, uftpd is as secure as any of its peers. While uftpd supports both FTP and TFTP, uftpd's author points to research that deems the latter insecure for use over the internet.

One of the main motivations behind uftpd, in its author's own words, was the complexity of the configuration files of the other available options. So in the spirit of open source software, he wrote uftpd to scratch his own itch: an FTP server without a configuration file, which you can use pretty much straight after installation without wading through a tonne of options.

While the project isn't available in the software repositories of most distributions, the installation from source is well-documented on uftpd's website and is really straightforward. Once it's installed, you can use uftpd as a regular user, the root user and even from a chroot environment, if needed.

Pros

Can be used right after installation without spending time with configuration files.

Cons

To use it you'll have to compile it manually, which isn't difficult but it is a little inconvenient.

Great for...

Making files available over FTP without much effort, as long as security isn't vital.

http://troglobit.com/projects/uftpd

SYSTEM OPTIMISER

Stacer 1.0.9

A graphical system monitor and resource optimiser

System monitoring is an essential task for any user. Although you'll find several system monitors in distribution repositories, Stacer offers the added benefit of an attractive user interface that's intuitive to navigate. Besides monitoring, you can also use the app to optimise and fine-tune several aspects of your installation – much like Bleachbit, but with a much nicer interface.

Stacer has a PPA repository for Ubuntu and also hosts precompiled RPM and DEB files in addition to an AppImage for other distributions. The app's dashboard displays information about the system and has dials to depict the activity levels of the processor, memory and disk, along with bars for download and upload speeds. You can also configure the app to alert you in case the readings on any of the dials exceeds a defined level.

Use the icons on the left to switch to the different components/areas of the app. You can add and disable startup apps and control other installed services. The app also helps you free up disk space by removing unwanted files from particular locations, such as the Trash and package caches. In the same vein, it can also display a list of all installed apps and enables you to uninstall any of them.

Above You can use Stacer to modify some of GNOME 3's quirks, like missing desktop icons

Pros

An attractive-looking, intuitive app to monitor and tune several aspects of your installation.

Cons

Limited set of monitoring capabilities that don't go much beyond the very basic options.

Great for...

Clearing package caches and controlling services with the minimum of fuss.

http://bit.ly/lud_stacer

Get your listing in our directory
To advertise here, contact Chris
chris.mitchell@futurenet.com | +44 01225 68 7832 (ext. 7832)

RECOMMENDED

Hosting listings

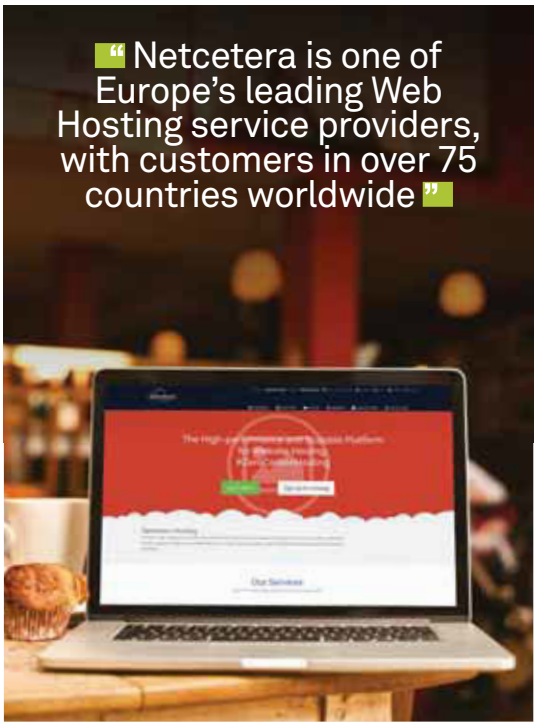
Featured host:
www.netcetera.co.uk
03330 439780

About us

Formed in 1996, Netcetera is one of Europe's leading web hosting service providers, with customers in over 75 countries worldwide. It is a leading IT infrastructure provider offering co-location, dedicated servers and managed infrastructure services to businesses worldwide.

What we offer

- Managed Hosting
  A full range of solutions for a cost-effective, reliable, secure host
- Dedicated Servers
  Single server through to a full racks with FREE setup and a generous bandwidth allowance
- Cloud Hosting
  Linux, Windows, hybrid and private cloud solutions with support and scaleability features
- Datacentre co-location from quad-core up to smart servers, with quick setup and full customisation



Five tips from the pros

01 **Optimise your website images**
When uploading your website to the internet, make sure all of your images are optimised for the web. Try using **jpegmini.com** software; or if using WordPress, install the EWWW Image Optimizer plugin.

02 **Host your website in the UK**
Make sure your website is hosted in the UK, and not just for legal reasons. If your server is located overseas, you may be missing out on search engine rankings on google.co.uk – you can check where your site is based on **www.check-host.net**.

03 **Do you make regular backups?**
How would it affect your business if you lost your website today? It's vital to always make your own backups; even if your host offers you a backup solution, it's important to take responsibility for your own data and protect it.

04 **Trying to rank on Google?**
Google made some changes in 2015. If you're struggling to rank on Google, make sure that your website is mobile-responsive. Plus, Google now prefers secure (HTTPS) websites. Contact your host to set up and force HTTPS on your website.

05 **Avoid cheap hosting**
We're sure you've seen those TV adverts for domain and hosting for £1! Think about the logic... for £1, how many clients will be jam-packed onto that server? Surely they would use cheap £20 drives rather than £1k+ enterprise SSDs? Remember: you do get what you pay for.

Testimonials

David Brewer
"I bought an SSL certificate. Purchasing is painless, and only takes a few minutes. My difficulty is installing the certificate, which is something I can never do. However, I simply raise a trouble ticket and the support team are quickly on the case. Within ten minutes I hear from the certificate signing authority, and approve. The support team then installed the certificate for me."

Tracy Hops
"We have several servers from Netcetera and the network connectivity is top-notch – great uptime and speed is never an issue. Tech support is knowledge and quick in replying – which is a bonus. We would highly recommend Netcetera."

J Edwards
"After trying out lots of other hosting companies, you seem to have the best customer service by a long way, and all the features I need. Shared hosting is very fast, and the control panel is comprehensive..."

Supreme hosting

CWCS MANAGED HOSTING — SUPREME HOSTING. SUPREME SUPPORT.

www.cwcs.co.uk
0800 1 777 000

CWCS Managed Hosting is the UK's leading hosting specialist. It offers a fully comprehensive range of hosting products, services and support. Its highly trained staff are not only hosting experts, it's also committed to delivering a great customer experience and is passionate about what it does.

- Colocation hosting
- VPS
- 100% Network uptime

Value hosting

elastichosts

elastichosts.co.uk
02071 838250

ElasticHosts offers simple, flexible and cost-effective cloud services with high performance, availability and scalability for businesses worldwide. Its team of engineers provide excellent support around the clock over the phone, email and ticketing system.

- Cloud servers on any OS
- Linux OS containers
- World-class 24/7 support

Small business host

HOSTPAPA

www.hostpapa.co.uk
0800 051 7126

HostPapa is an award-winning web hosting service and a leader in green hosting. It offers one of the most fully featured hosting packages on the market, along with 24/7 customer support, learning resources and outstanding reliability.

- Website builder
- Budget prices
- Unlimited databases

Enterprise hosting:

2020Media www.2020media.com

www.2020media.com | 0800 035 6364

WordPress comes pre-installed for new users or with free managed migration. The managed WordPress service is completely free for the first year.

We are known for our "Knowledgeable and excellent service" and we serve agencies, designers, developers and small businesses across the UK.

Budget hosting:

HETZNER ONLINE

www.hetzner.de/us | +49 (0)9831 5050

Hetzner Online is a professional web hosting provider and experienced data-centre operator. Since 1997 the company has provided private and business clients with high-performance hosting products, as well as the necessary infrastructure for the efficient operation of websites. A combination of stable technology, attractive pricing and flexible support and services has enabled Hetzner Online to continuously strengthen its market position both nationally and internationally.

- Dedicated and shared hosting
- Colocation racks
- Internet domains and SSL certificates
- Storage boxes

SSD web hosting

bargainhost

www.bargainhost.co.uk
0843 289 2681

Since 2001, Bargain Host has campaigned to offer the lowest-priced possible hosting in the UK. It has achieved this goal successfully and built up a large client database which includes many repeat customers. It has also won several awards for providing an outstanding hosting service.

- Shared hosting
- Cloud servers
- Domain names

Value Linux hosting

PATCHMAN WEB HOSTING

patchman-hosting.co.uk
01642 424 237

Linux hosting is a great solution for home users, business users and web designers looking for cost-effective and powerful hosting. Whether you are building a single-page portfolio, or you are running a database-driven ecommerce website, there is a Linux hosting solution for you.

- Student hosting deals
- Site designer
- Domain names

Fast, reliable hosting

:BYTEMARK

www.bytemark.co.uk
01904 890 890

Founded in 2002, Bytemark are "the UK experts in cloud & dedicated hosting". Its manifesto includes in-house expertise, transparent pricing, free software support, keeping promises made by support staff and top-quality hosting hardware at fair prices.

- Managed hosting
- UK cloud hosting
- Linux hosting

Free Resources

Welcome to Filesilo!

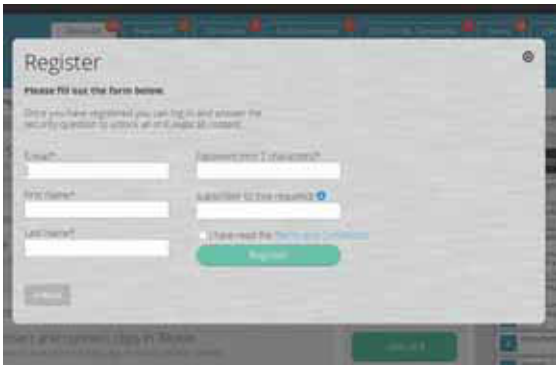
Download the best distros, essential FOSS and all our tutorial project files from your FileSilo account

WHAT IS IT?

Every time you see this symbol in the magazine, there is free online content that's waiting to be unlocked on FileSilo.

WHY REGISTER?

- Secure and safe online access, from anywhere
- Free access for every reader, print and digital
- Download only the files you want, when you want
- All your gifts, from all your issues, all in one place



1. UNLOCK YOUR CONTENT

Go to **www.filesilo.co.uk/linuxuser** and follow the instructions on screen to create an account with our secure FileSilo system. When your issue arrives or you download your digital edition, log into your account and unlock individual issues by answering a simple question based on the pages of the magazine for instant access to the extras. Simple!

2. ENJOY THE RESOURCES

You can access FileSilo on any computer, tablet or smartphone device using any popular browser. However, we recommend that you use a computer to download content, as you may not be able to download files to other devices. If you have any problems with accessing content on FileSilo, take a look at the FAQs online or email our team at **filesilohelp@futurenet.com**.

Free for digital readers too!
Read on your tablet, download on your computer

Log in to **www.filesilo.co.uk/linuxuser**

Subscribe and get instant access

Get access to our entire library of resources with a money-saving subscription to the magazine – subscribe today!

This month find...

DISTROS

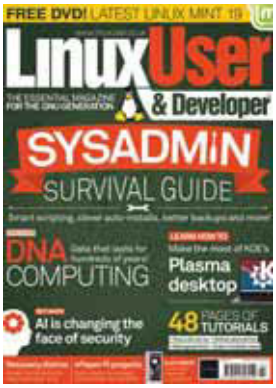
Linux Mint 19, 'Tara', is the latest long term release of the hugely popular distribution. PCLinuxOS is another distro with strong community support, while Devuan 2.0 for the Raspberry Pi rounds off the list.

SOFTWARE

To accompany our group test on system recovery distros, we've gathered some of the best as a System Rescue and Repair Bundle – including the fabulous Ultimate Boot CD and GParted Live.

TUTORIAL CODE

All the code for the Azure PowerShell tutorial, plus computer security.



Subscribe & save!

See all the details on how to subscribe on **page 30**

FOLLOW US
Facebook: facebook.com/LinuxUserUK
Twitter: @linuxusermag

NEAR-FUTURE FICTION

Obnoxious

ABOUT
Stephen Oram
Stephen writes near-future science fiction. He's been a hippie-punk, religious-squatter and a bureaucrat-anarchist; he thrives on contradictions. He has publised two novels, *Quantum Confessions* and *Fluence*, and is in several anthologies. His recent collection, *Eating Robots and Other Stories*, was described by the *Morning Star* as one of the top radical works of fiction in 2017. **http://stephenoram.net**

It's embarrassing,' said Nialle as he put his empty coffee cup down.

She picked up his cup and put it in the sink. 'Yeah, but it's a seriously cool jacket,' she said.

He shrugged and lifted the wearable tech jacket from its hook. He turned to kiss her goodbye, brushing the hair from in front of her face and kissing her hand by mistake as she also tried to brush it away.

She chuckled. 'You'll be glad of it once you're on that crowded train.'

She was right, as always, and he knew it. But, that didn't make him cringe any less at the thought of using it. He shouted as he closed the front door, 'See you tonight.'

She shouted something back, but he didn't catch it.

At the station, as she'd predicted, the platform was crammed full of commuters standing patiently. A halo of irritation and early morning grumpiness hung silently above them.

He stepped into a tiny space between two headphone-wearing men. They shuffled a little further forward and, although it was less than half a step, they still felt the need to scowl and make their annoyance known.

The train was due, and slowly but surely they were nudged closer and closer to each other as more commuters arrived. He'd become expert over the past few months at judging the number of people on the platform and how many of them could fit onto a train. He looked around; he reckoned he would just about get on.

A man to his left took a step backwards as the train appeared from around the corner. Immediately the crowd pushed him back into his place; there was no room for slacking at this stage of the game.

The doors beeped and opened. The crowd moved forward gently and politely, but the undercurrent of competition was palpable. He edged forward in unison with his fellow sufferers. He could feel the tension rising around him as the train seemed to be filling up quicker than they were moving forward. The man in front of him stepped on and paused in the doorway. Nialle could see some space in front of the stalled man so he grabbed the handrail and pulled himself up, one foot in and one foot out.

> "The doors beeped and opened. The crowd moved forward gently and politely, but the undercurrent of competition was palpable"

He touched the man's back and he moved just enough for Nialle to get both feet onto the train. The doors beeped and closed behind him. It was a crush, but he knew that the seasoned travellers would have made themselves slightly larger when everyone was trying to board so they could now relax and create themselves some precious space.

This was his cue.

He pinched the corner of his jacket collar.

The crowd near him shuffled a little and started to move away. Hostile looks were thrown his way, and there was much tutting and some occasional under-the-breath swearing.

The wearable technology was emitting a strong scent of stale body odour, creating enough space around him to make the embarrassment worth it.

He smiled; once again, his jacket had done its job. ■

NEXT ISSUE **ON SALE 23 AUGUST**
Open Source Storage | Purism's mobile OSes | Firewalls

techradar.

The source for tech buying advice

techradar.com

The Electronic Frontier Foundation is the leading nonprofit organization defending civil liberties in the digital world. Founded in 1990, EFF champions user privacy, free expression, and innovation through impact litigation, policy analysis, grassroots activism, and technology development. We work to ensure that rights and freedoms are enhanced and protected as our use of technology grows.

EFF.ORG

ELECTRONIC FRONTIER FOUNDATION

Protecting Rights and Promoting Freedom on the Electronic Frontier